PHP

Améliorez la performance de vos applications

http://www.free-livres.com/

Jean-Marie RENOUARD

Résumé

Ce livre traite du langage **PHP en versions 5.3 et supérieure**. Il s'adresse aux **développeurs PHP et intégrateurs.** Il détaille les aspects techniques et architecturaux inhérents à tout projet de développement avec pour objectif d'atteindre la **pérennité** et **l'efficacité** au quotidien. Une bonne connaissance des bases du langage ainsi que l'expérience de l'écriture de scripts en PHP et MySQL sont des pré-requis souhaitables pour tirer le meilleur profit de ce livre. Ce livre permet aussi aux **acteurs techniques d'un projet Web** ayant découvert la programmation PHP côté serveur de mieux cerner les problématiques techniques telles que la **sécurisation** et **l'optimisation des environnements techniques**.
À la fin de ce livre, vous pourrez prétendre mieux connaître le langage PHP et son **environnement de déploiement** et vous serez capable de mettre en place des **stratégies rapides** et **efficaces** pour la gestion de votre code PHP, son **évolutivité** et surtout pour la **qualité de fonctionnement** en situation de production, c'est-à-dire sur une plate-forme pouvant recevoir de nombreux utilisateurs simultanément et devant fournir des résultats rapides et fiables.
Du point de vue de la réalisation d'applications vous aurez une vue plus claire des **architectures logicielles**, c'est-à-dire de la manière d'organiser et de structurer l'ensemble du code de votre projet afin de pouvoir garantir **l'évolutivité** et le travail en équipe de manière performante et optimale.
Le code source des exemples du livre est en téléchargement sur le site www-editions-eni.fr.

Les chapitres du livre :
Introduction – Les nouveautés du PHP 5.3 – Sécurisation d'un serveur PHP – Sécurisation du code PHP – Optimisation de l'exécution du code PHP – Qualité du code PHP – Architecture logicielle et PHP objet – Patrons de conception utiles en PHP – Assemblage vs développement – Architecture haute disponibilité pour le PHP – Gestion de code PHP en production – Management de la qualité de la plate-forme

L'auteur

Jean-Marie RENOUARD est Conseiller en architecture web. Ses différentes missions lui permettent d'intervenir sur l'administration de bases de données, l'intégration technique applicative et d'évoluer sur des environnements Web à haute disponibilité. À travers les pages de ce livre il fait partager au lecteur toute son expérience du développement avancé sous PHP.

Ce livre numérique a été conçu et est diffusé dans le respect des droits d'auteur. Toutes les marques citées ont été déposées par leur éditeur respectif. La loi du 11 Mars 1957 n'autorisant aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les "copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective", et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, "toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayant cause, est illicite" (alinéa 1er de l'article 40). Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal. Copyright Editions ENI

Ce livre numérique intègre plusieurs mesures de protection dont un marquage lié à votre identifiant visible sur les principales images.

Avant-propos

Ce livre traite du langage PHP en version 5.3 et supérieure. Il traite des aspects techniques et architecturaux inhérents à tout projet informatique visant la pérennité et l'efficacité au quotidien tout en garantissant une approche de qualité. Tous les principes présentés sont illustrés à l'aide de cas concrets.

© ENI Editions - All rigths reserved - Algeria Educ

Qu’est-ce que le langage PHP ?

Le langage PHP est un langage de programmation interprété utilisé spécifiquement pour l’écriture de sites web dynamiques. L’interpréteur PHP repose sur le moteur Zend 2 écrit en langage C. L’interpréteur PHP est aussi très bien intégré dans le serveur Web Apache sous forme de module et facilite son installation et son utilisation en environnement Web.

Le PHP se caractérise par sa facilité de programmation et par sa structure de données tableau, rapide à comprendre et à utiliser.

```
<?php
// tableau classique indexé par entier
$tableau=array( « V1 », « v2 », « v3 », 1, 45, ’a’ );

echo $tableau[0]; // « V1 »

// boucle
foreach($tableau as $element) {
    echo $element;
}

// tableau indexé par clé
$table=array( « V1 » => 4, « v2 » => 5, « v3 »=>100 );

echo $table["v2"]; // 5

// boucle
foreach($tableau as $cle => $valeur) {
    echo "$key => $valeur";
}
?>
```

Faiblement typé, le langage PHP utilise le mécanisme d’inférence, c’est-à-dire la recherche automatique du type en fonction des expressions associées.

```
<?php
$chaine= « une chaine » ; // chaîne de caractère déduite par inférence
$entier=1 ; // entier déduit
$float=1.23 ; // flottant déduit
?>
```

Le langage PHP arrive tardivement dans le sillage de langages tel que C, Perl ou Java (auquel il emprunte largement la syntaxe). Il garantit aux nouveaux venus ayant déjà pratiqué l’un de ces langages de programmation de rapidement monter en compétence et d’obtenir aisément leurs premiers programmes en PHP.

© ENI Editions - All rigths reserved - Algeria Educ

L’histoire courte du PHP

Lors de sa première sortie, le 8 juin 1995, la syntaxe de PHP/FI 1.0 était assez éloignée de la syntaxe actuelle. D’ailleurs, l’acronyme PHP n’avait pas son sens actuel (PHP/FI : *Personal Home Page/Form Interpreter*). Son créateur Rasmus Lerdorf va même le considérer en novembre 1997, lors de la sortie de la première version, comme l’outil le plus rapide et le plus simple pour la création de site web dynamique.

La version 3 du langage va faire basculer l’évolution du langage dans un modèle de développement plus communautaire. En effet, en juin 1998, de nombreux développeurs et surtout deux Israéliens, Zeev Suraski et Andi Gutmans, vont jeter les bases de la nouvelle syntaxe du langage. C’est d’ailleurs lors de la sortie de cette nouvelle version que PHP sera rebaptisé « PHP : Hypertext Preprocessor » et offrira une API permettant d’étendre et d’ajouter de nouvelles fonctions rapidement.

Deux ans plus tard, en juin 2000, la version 4.0 voit émerger le moteur de PHP, le moteur Zend, dont le nom est la concaténation du nom des deux principaux développeurs nommés plus haut. La version 4, dans la lignée de la version 3, va connaître un vrai engouement lié à son intégration parfaite avec le moteur web Apache et sera massivement déployée chez tous les hébergeurs de sites Web gratuits ou payants. Cette version va subsister pendant un cycle de vie de quasiment huit années. Elle connaîtra six sous-versions, apportant la stabilité et des performances acceptables pour la réalisation de pages web dynamiques.

L’évolution du PHP arrive naturellement en juillet 2004 avec un support avancé des concepts objets. La syntaxe objet va évoluer et un nouveau moteur, le moteur Zend 2, va voir le jour et propulser PHP dans l’ensemble des langages ayant une véritable capacité à supporter les développements orientés objet.

Depuis, le langage PHP version 5, seule version maintenue actuellement, a connu une petite dizaine de sous-versions, souvent pour améliorer la performance et la sécurité de l’ensemble.

La version 5.3 de juin 2009 peut être considérée comme une version très importante car elle apporte une évolution de la syntaxe avec le support des labels et des fonctions anonymes ou fonctions lambda, le support des espaces de nommage et l’implémentation d’un ramasse-miettes, outil idéal pour garantir la stabilité des programmes résidant en mémoire pour des tâches en traitement batch par exemple.

L’évolution du langage PHP est très importante et le cycle de développement semble plus court et les évolutions plus importantes aujourd’hui. Ceci s’explique aussi par l’ampleur de l’utilisation de PHP et la popularité très grande de nombreuses applications développées en langage PHP telles que Drupal, PHPunit, Smarty, PHPList, SugarCRM, phpWebGallery, Mantis et surtout la masse importante de scripts, de classes et d’exemples d’utilisation de PHP accessibles en quelques clics sur Internet.

© ENI Editions - All rigths reserved - Algeria Educ

La licence PHP

La licence PHP est actuellement considérée comme une licence libre par l'OSI (*Open source Initiative*), organisation de proposition des logiciels libres chargée de recenser les licences logiciels respectant les quatre libertés :

- Liberté d'utilisation.
- Liberté d'étudier le logiciel.
- Liberté de modifier le logiciel.
- Liberté de redistribuer le logiciel et ses modifications.

La fondation du logiciel libre (FSF) considère cependant que le PHP n'est pas compatible avec la Licence GNU GPL, licence libre la plus utilisée dans le monde car l'acronyme PHP comporte des restrictions d'utilisation.

Nous pouvons donc cependant confirmer que la licence PHP est une licence respectant pleinement les libertés des utilisateurs et, à ce titre, doit être considérée comme une licence libre comme l'a fait l'OSI.

Le langage PHP ainsi que le moteur PHP sont donc des logiciels libres et la popularité de PHP n'a donc pas été entravée par l'esprit des pionniers du langage, bien au contraire.

© ENI Editions - All rigths reserved - Algeria Educ

À qui s’adresse ce livre ?

Ce livre s’adresse à des lecteurs ayant déjà acquis les bases du langage PHP et ayant écrit plusieurs scripts en PHP et MySQL. Ces lecteurs auront la possibilité d’élargir leur connaissance du langage et des différents concepts du développement orienté objet.

Ce livre permet aussi aux maquettistes Web ayant mis un pied dans la programmation PHP côté serveur de mieux cerner les problématiques telles que la sécurisation et l’optimisation des environnements techniques.

Aucune connaissance particulière du CSS et du JavaScript n’est nécessaire ou requise pour la lecture de cet ouvrage.

Au travers du concept objet et de son expression spécifique dans le langage PHP, chaque lecteur aura la possibilité de mieux comprendre les enjeux majeurs des technologies serveur et de trouver des réponses adaptées et viables.

À la fin de ce livre, vous pourrez prétendre mieux connaître le langage PHP et son environnement de déploiement.

Vous serez capable de mettre en place des stratégies rapides et efficaces pour la gestion de votre code PHP, pour son évolutivité et surtout pour la qualité de fonctionnement en situation de production, c’est-à-dire sur une plate-forme pouvant recevoir de nombreux utilisateurs simultanément et devant fournir des résultats rapides et fiables.

Du point de vue de la réalisation d’application, vous aurez une vue plus claire des architectures logicielles, c’est-à-dire la manière d’organiser et structurer l’ensemble du code de votre projet afin de pouvoir garantir l’évolutivité et le travail en équipe de manière performante et optimale.

© ENI Editions - All rigths reserved - Algeria Educ

Que contient ce livre ?

Ce livre comporte 11 chapitres permettant d'aborder l'environnement et les différents aspects de PHP.

1. Les nouveautés de PHP 5.3

Ce premier chapitre contient l'ensemble des informations relatives aux nouvelles fonctionnalités des versions 5 et 5.3 du langage PHP. Il couvre la plupart des fonctionnalités majeures des évolutions et vous permet de mieux comprendre pourquoi il est important de passer votre code vers la dernière version de PHP.

Ce chapitre met l'accent sur la syntaxe des nouveautés en prenant le parti que vous connaissez déjà soit un autre langage de programmation, soit la version 4 du langage PHP.

2. Sécurisation d'un serveur PHP

Ce deuxième chapitre est consacré aux concepts fondamentaux de la sécurité des systèmes hébergeant vos applications. Il permet de mieux comprendre comment les serveurs hébergeant vos applications PHP sont fragilisés, soit par négligence, soit par méconnaissance.

Ce chapitre met l'accent sur les moyens simples et rapides de dissimuler ou de maquiller des informations techniques sur votre serveur, votre application et sur la technologie sous-jacente, c'est-à-dire les versions de PHP, de MySQL ou de votre système d'exploitation utilisées pour héberger votre application PHP.

3. Sécurisation du code PHP

Ce troisième chapitre est consacré aux concepts fondamentaux de la sécurité des applications PHP et plus particulièrement du code en lui-même. Il permet de mieux comprendre comment les applications sont vulnérables. Non pas faute de sécurisation de l'environnement système et réseau, mais par de bonnes pratiques dans votre code qui ne sont pas mises en œuvre (faute de temps ou bien à cause d'une croyance souvent erronée affirmant que les failles de sécurité sont peu risquées).

Ce chapitre met en évidence des cas d'attaques typiques et les solutions techniques de base à mettre en œuvre pour couper l'ensemble de ces attaques sur le Web.

4. Optimisation de l'exécution du code PHP

Ce quatrième chapitre est consacré aux bases d'optimisation des traitements PHP. Est abordé dans ce chapitre, l'ensemble des optimisations possibles sur un serveur Apache 2.2 en termes de paramétrage fin et d'adaptation à votre population d'utilisateur. Ensuite, sont présentées les stratégies de test et de mesure de la performance afin de détecter et surtout de cibler les sources majeures de dégradation des performances. Enfin, cinq solutions d'optimisation sont proposées pour améliorer rapidement les temps de traitement et de rendu de votre résultat.

5. Qualité du code PHP

Ce cinquième chapitre est consacré à une démarche de qualité intégrée aux développements basés sur le principe de développement piloté par les tests. Dans ce chapitre est présenté l'ensemble des arguments et des cas concrets de réalisation pilotée par les tests afin de souligner l'efficacité de ce type d'approche dans la maintenance à long terme de vos applications.

Pour illustrer les exemples présentés, nous utilisons le composant PHPUnit 3.

6. Architecture logicielle et le PHP objet

Ce sixième chapitre aborde les bases des architectures orientées objet au sein des applications. Ce chapitre présente la plus répandue des architectures logicielles : le MVC et son découpage en couche.

Dans ce chapitre, nous revenons rapidement sur l'importance de bien concevoir une architecture logicielle en se basant sur des techniques de modélisation standard telles que l'UML afin de garantir le bon déroulement du projet.

© ENI Editions - All rigths reserved - Algeria Educ

Dans ce chapitre, vous aborderez les techniques de traduction de modèle permettant d'obtenir du code fonctionnel et quasiment prêt à fonctionner à partir de vos modèles UML.

7. Les patrons de conception utiles en PHP

Le septième chapitre vous offre la possibilité de mieux percevoir l'importance et la relation intime entre modèle UML et code source PHP. Ce chapitre totalement complémentaire avec le précédent vous dévoile un certain nombre d'agencements de classe UML (appelés communément patrons de conception ou design patterns). Il s'agit de modèles génériques préétablis permettant de répondre rapidement à des problèmes de conception courants et offrant une solution structurelle élégante, adaptative et simple. Ce chapitre vous offre la possibilité d'explorer une dizaine de patrons et de résoudre des problèmes de conception qui ont amené de nombreux projets à faire le choix de solutions lourdes et peu maintenables.

8. Assemblage vs Développement

Dans ce huitième chapitre est abordée la problématique qui oppose actuellement les développeurs pure souche ayant pour principe de comprendre et de maîtriser le code source qu'ils utilisent et les partisans d'une vision plus simple qui consiste à embarquer tout composant répondant grossièrement au besoin actuel d'un projet. Ce chapitre a pour but de tenter la grande conciliation en mettant en avant les techniques et stratégies pour maîtriser son code source et celui des autres que l'on assemble dans notre propre projet. Ce chapitre parle sans détour des avantages de l'intégration de composants fiables et bien testés et propose un ensemble de sources permettant de trouver votre bonheur dans la large base de code ouvert sur Internet sans tomber dans les pièges classiques de ce type d'approche.

9. Architecture haute disponibilité pour le PHP

Le neuvième chapitre met en avant quelques-unes des nombreuses techniques utilisées pour convertir vos applications PHP pour la haute disponibilité. La haute disponibilité n'ayant de sens que quand elle est implémentée de bout en bout, c'est-à-dire de l'accès réseau en passant par la redondance des serveurs Web Apache et des serveurs de bases de données (chacune des briques devant être au moins doublée pour garantir la haute disponibilité).

10. Gestion de code PHP en production

Ce dixième chapitre évoque l'ensemble des problématiques de gestion de code PHP en production. Y sont évoquées les différentes solutions de packaging des applications tel que RPM ou, encore mieux, la solution native de PHP, les archives Phar. Ce chapitre aborde également les problématiques de stratégie de déploiement de gestion de la configuration et des éventuels retours en arrière en cas de problème.

11. Management de la qualité de la plate-forme

Ce onzième et dernier chapitre traite d'un sujet important dans la pérennité et le succès d'une application Web en PHP : la qualité.

Nous voyons comment mettre en place un système simple de gestion de la qualité basé sur les travaux de Deming. Ce chapitre explore aussi l'ensemble des techniques efficaces de collecte et de mesure d'indicateur sur vos applications. Enfin, est présenté un système efficace de collecte des informations et de traitement des informations concernant les problèmes et la gestion des traces applicatives dans un système de production.

© ENI Editions - All rigths reserved - Algeria Educ

Où trouver de l’aide ?

Ce livre est écrit par Jean-Marie Renouard qui peut être joint pour toute question à l’e-mail suivant : jmrenouard@gmail.com.

Il sera ravi de répondre ou tenter de trouver un début de réponse à vos questions concernant ce très beau et très dynamique langage de programmation qu’est le PHP.

© ENI Editions - All rigths reserved - Algeria Educ

Les avancées de la version 5 du langage PHP

1. La version tant attendue : PHP5

La version 5 du langage PHP est arrivée avec un nouveau moteur d'interprétation : le moteur Zend 2. Son cycle de développement va durer plus d'un an (du 29 juin 2003 au 14 juillet 2004), entre la première version de développement et la version stable ou release 5.0.0.

Le langage PHP 5 apporte d'abord des améliorations et des nouvelles fonctionnalités telles que :

- Amélioration de la gestion des flux réseau.
- Inclusion du support IPv6.
- Amélioration du support GD.
- Amélioration de la consommation mémoire.

Il se caractérise aussi par des changements majeurs traduits par trois axes principaux :

- Syntaxe enrichie pour le support des notions objet.
- Ajout du support de structure complexe de données.
- Gestion des données SQL et XML améliorée.

Voyons maintenant plus en détail chacun de ces trois aspects du nouveau PHP 5.

2. Le nouveau modèle objet pour PHP

a. Nouvelle syntaxe et possibilités objet

L'apparition d'une syntaxe enrichie pour le support de l'ensemble des concepts objets a propulsé PHP sur le devant de la scène des langages de programmation en offrant aux développeurs la capacité de coder des programmes orientés objet avec des fonctionnalités telles que :

- Gestion de l'encapsulation des propriétés.
- Gestion des exceptions.
- Apparition des fonctions magiques comme les constructeurs et les destructeurs.
- Support des notions d'interface et d'éléments abstraits.
- Possibilité partielle de validation de type pour les paramètres.
- Apparition d'une API complète d'introspection des objets et des classes.

Il devient facile d'exprimer rapidement l'ensemble des modèles classiques objets tel que les relations, les associations ou encore les héritages. Mais plus que cela, le langage PHP 5 permet de fournir des implémentations simples pour l'ensemble des patrons de conception chers aux architectes logiciel qui n'ont pas trouvé, dans les versions précédentes, la consistance syntaxique suffisante pour réaliser une implémentation fidèle aux auteurs de ces patrons. Nous aborderons ces patrons de conception objet dans un chapitre qui leur est dédié.

Il faut noter que la syntaxe objet du langage PHP a largement été influencée par le langage Java, le C++ et le langage Python. Dans le même esprit, PHP limite les mécanismes d'héritage complexe et lourd directement au

© ENI Editions - All rigths reserved - Algeria Educ

travers de la syntaxe du langage PHP.

Voici donc une courte présentation des grands aspects de la programmation objet en PHP 5.x.

b. Les classes et les méthodes abstraites

Le langage PHP 5 voit apparaître un nouveau mot clé : `abstract`.

Ce mot clé peut être appliqué à deux types d'éléments du langage : les classes et les méthodes.

La syntaxe d'une classe abstraite est donc :

```
abstract class Democratie {
...
}
```

La syntaxe d'une méthode abstraite est donc :

```
abstract public maMethodeAbstraite() {
...
}
```

Il faut noter qu'une méthode abstraite ne peut être déclarée qu'à l'intérieur d'une classe elle-même abstraite.

c. Les interfaces

Une interface est l'équivalent d'une classe abstraite pure, c'est-à-dire d'une classe n'ayant aucun attribut et composée uniquement de méthodes abstraites.

La syntaxe de déclaration est simple et consiste à changer le mot clé `abstract class` par le mot clé `interface` :

```
abstract class véhicule {
...
}
```

Devient :

```
interface véhicule {
...
}
```

La syntaxe d'utilisation aussi est simple et consiste à changer le mot clé `extends` par le mot clé `implements` :

```
class Voiture extends Vehicule{
...
}
```

Devient :

```
class Voiture implements Vehicule {
...
}
```

Exemple d'une interface Véhicule

```
< ?php
interface Vehicule {
    public function avancer() ;
    public function reculer() ;
    public function tournerAGauche() ;
    public function tournerADroite() ;
    public function accelerer() ;
    public function ralentir() ;
}
```


© ENI Editions - All rigths reserved - Algeria Educ

```
?>
```

Exemple d'utilisation de l'interface

```
< ?php
class Voiture inplements Vehicule {
    public function avancer()  { echo « Avancer » ; }
    public function reculer() { echo « Reculer » ; }
    public function tournerAGauche() { echo « Tourner A Gauche » ; }
    public function tournerADroite() { echo « Tourner A Droite » ; }
    public function accelerer() { echo « Accelerer » ; }
    public function ralentir() { echo « Ralentir » ; }
}
?>
```

Il est à noter que toutes les fonctions de l'interface doivent être définies dans la classe qui « implémente » ou « hérite » de l'interface.

d. Contrôle d'accès aux propriétés d'un objet

Le contrôle d'accès aux propriétés d'une classe est une nouveauté permettant de supporter le principe d'encapsulation au travers des trois types d'accès classiques du modèle objet :

Opérateurs d'accès	Description
public	L'accès à l'attribut est autorisé par l'ensemble des classes, objets ou scripts extérieurs à la classe ou à l'objet.
protected	L'accès à la propriété n'est permis que depuis l'intérieur de la classe, depuis une classe dérivée ou d'une classe parent.
private	L'accès n'est autorisé que depuis la classe elle-même.

La portée d'une propriété est toujours le premier élément de la déclaration d'une propriété.

Si le mot clé var est utilisé sans portée explicite alors la portée publique s'applique par défaut.

Exemple de déclaration de portée dans une classe

```
< ?php
class GestionPortee {
    // Attributs
    public $attributPublic ;
    protected $attributProtege ;
    private $attributPrive ;

    // Attributs statiques
    public static $attributPublicStatique ;
    protected static $attributProtegeStatique ;
    private static $attributPriveStatique ;

    // Méthodes
    public function fonctionPublic() ;
    protected function fonctionProtege() ;
    private function fonctionPrive() ;

    // Méthodes statiques
    public static function fonctionPublicStatique () ;
    protected static function fonctionProtegeStatique() ;
    private static function fonctionPriveStatique() ;
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

e. Validation de type

Il est possible de valider le type des paramètres passés aux méthodes d'une classe ainsi que le type des paramètres passés aux fonctions PHP. La seule limite de cette fonctionnalité est qu'elle n'est applicable qu'aux objets et aux tableaux depuis la version 5.1.

La validation de type s'appuie sur une déclaration plus précise de chaque fonction ou méthode afin de déclarer le type attendu par la fonction ou la méthode.

Voici donc deux exemples permettant de mettre en place rapidement ce mécanisme dans votre code :

Fonction demandant un tableau en paramètre

```
<?php
function compteTableau (array $tableau) {
    return count($tableau);
}
?>
```

Résultat d'un passage de mauvais paramètre

```
Echo « Nb element ». compteTableau(« r ») ;
```

Un mauvais passage de paramètre produit une exception :

```
Catchable fatal error: Argument 1 passed to compteTableau() must
be an array, string given, called in validation2.php on line 7 and
defined in validation2.php on line 3
```

Fonction demandant un objet de la classe Utilisateur en paramètre

```
<?php
class Utilisateur {
    private $pseudo;
    public function __construct($p='') {
        $this->pseudo=$p;
    }
    public function getPseudo() {
        return $this->pseudo;
    }
}

function estBonUtilisateur(Utilisateur $u, $pseudo) {
    if ($u->GetPseudo()==$pseudo) return true;
    return false;
}
?>
```

Résultat d'un passage de mauvais paramètre

```
$titof=new Utilisateur('titof');

if (estBonUtilisateur( "titi", "titof")) echo "Bizarre";
else echo "Cela pourrait se comprendre";
```

Un mauvais passage de paramètre produit une exception :

```
Catchable fatal error: Argument 1 passed to estBonUtilisateur()
must be an instance of Utilisateur, string given, called in
validation.php on line 23 and defined in validation.php on line 16
```

f. Apparition des fonctions magiques

© ENI Editions - All rigths reserved - Algeria Educ

Avec l'avènement de cette version et de ces versions correctives apparaissent un certain nombre de fonctions magiques car elles sont invoquées dans des configurations particulières et permettent d'enrichir le comportement de la classe dans ces circonstances.

Voici le tableau des fonctions magiques :

Fonctions	Description
__autoload($nomDeLaClasse)	Méthode d'interception des appels inaccessibles sur une classe lors de l'instanciation de classe. Lorsque l'opérateur new échoue car la classe et sa définition n'existent pas, cette méthode est invoquée avec le nom de la classe en paramètre. Cette méthode est utilisée pour un chargement du code des classes en mode juste à temps c'est-à-dire que le chargement n'intervient qu'au premier appel de l'opérateur new.

Méthodes de classe	Description
__construct(…)	Constructeur de l'objet pouvant prendre différents paramètres qui seront récupérés et transmis lors de l'appel à l'opérateur new. `$u=new Utilisateur('titof') ;` Le paramètre 'titof' sera transmis à la méthode magique __construct.
__destruct()	Destructeur de l'objet. Ne prend pas de paramètre. Dernière méthode de l'objet appelée avant sa disparition complète.
__call($nomMethode, array $arams)	Méthode d'interception des appels inaccessibles sur une méthode inexistante. Cette méthode prend le relais et évite l'apparition d'erreurs bloquantes dans le code. La méthode __call possède 2 paramètres : • Le nom de la méthode. • Le tableau des paramètres.
__callStatic($nomMethode, array $arams)	Equivalent de __call pour les méthodes statiques inexistantes.
__get($nomAttr)	Méthode d'interception des lectures inaccessibles sur un attribut de l'objet protégé ou sur des attributs inexistants. La méthode __get prend un seul paramètre : le nom de l'attribut.
__set($nomAttr, $value)	Méthode d'interception des écritures inaccessibles sur un attribut de l'objet protégé ou sur des attributs inexistants. La méthode __set prend 2 paramètres : le nom de l'attribut et la valeur souhaitée pour l'affectation.
__isset($nomAttr) ;	Méthode d'interception des appels empty() ou isset() sur un attribut de l'objet protégé ou sur des attributs inexistants. La méthode __isset prend un paramètre : le nom de l'attribut.
__unset($nomAttr) ;	Méthode d'interception des appels unset () sur un attribut de l'objet protégé ou sur des attributs inexistants. La méthode __unset prend un paramètre : le nom de l'attribut.

© ENI Editions - All rigths reserved - Algeria Educ

__sleep()	Méthode invoquée systématiquement avant l'appel à la fonction serialize() et permettant de réaliser les traitements importants avant une opération de sérialisation.
__wakeup()	Méthode invoquée systématiquement après l'appel à la fonction unserialize() et permettant de réaliser les traitements importants après une opération de reconstruction d'objet.
__toString()	Méthode renvoyant une chaîne de caractères. Son rôle est de permettre de représenter l'objet sous forme de chaîne facilitant son utilisation comme paramètre des fonctions echo () et print().
__invoke($params)	Méthode d'interception des appels utilisant l'objet comme une fonction. La méthode __invoke() prend un paramètre : le paramètre d'appel.
__set_state(array $attributs)	Méthode d'interception des appels à fonction var_export(). La méthode var_export() renvoie une représentation d'une variable sous forme de chaîne de caractères valide et interprétable par PHP.
__clone()	Méthode d'interception des appels à fonction clone(). La méthode clone() est invoquée après clonage sur le nouvel objet créé.

Exemple d'utilisation des fonctions magiques

```
<?php
class Profil {
    private $pseudo;

    public function __construct($p='anonyme') {
        echo "\n* Constructeur de la classe: ".__CLASS__ . "
avec $p";
        $this->pseudo=$p;
    }
    public function __destruct() {
        echo "\n* Destructeur de la classe: ".__CLASS__;
        echo "\nManipulation de : ". $this;
        echo "\n";
    }
    public function __toString() {
        $chaine=__CLASS__;
        $chaine.= " [";
        $chaine.= "\n\t Pseudo: ";
        $chaine.= $this->pseudo;
        $chaine.= "\n]\n";
        return $chaine;
    }
}
//Construction d'un objet Profil 'anonyme'
$p1=new Profil();

//Construction d'un objet Profil 'Garfield'
$p2=new profil('Garfield');

//Affichage des objets dans une chaine
echo "\n\n\$p1 = $p1";
echo "\n\n";
echo "\$p2 = $p2";

//Appel explicite au destructeur de $p2
$p2=null;
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

Résultats d'exécution

```
* Constructeur de la classe: Profil avec anonyme
* Constructeur de la classe: Profil avec Garfield

$p1 = Profil [
    Pseudo: anonyme
]


$p2 = Profil [
    Pseudo: Garfield
]

* Destructeur de la classe: Profil
Manipulation de : Profil [
    Pseudo: Garfield
]


* Destructeur de la classe: Profil
Manipulation de : Profil [
    Pseudo: anonyme
]
```

g. Gestion des mécanismes d'exception

La gestion des exceptions est basée dans de nombreux langages sur le principe du bloc à surveiller et du bloc de traitement d'exception.

Le Langage PHP ne va pas échapper à la règle et offre la même syntaxe que la plupart des langages de programmation orientés objet actuels.

Syntaxe de base pour l'envoi d'une exception

```
throw new Exception(« message d'erreur ») ;
```

Syntaxe de base pour le traitement des exceptions

```
try {
        // bloc à surveiller
        // des exceptions peuvent être levées par le mécanisme précédent
}
catch ( Exception $e ) {
        //bloc de traitement des exceptions levées
        // $e peut être directement utilisé comme chaine de caratère
}
```

Seule la méthode magique __toString() de la classe Exception peut être surchargée par héritage pour créer vos exceptions personnalisées.

Exemple de programme générant des exceptions

```
function envoyerMonException($raison) {
        throw new MonException("Envoi de mon exception pour raison($raison)");
}
```

Exemple de programme interceptant l'exception

```
try {
    echo "Traitement 1\n";
```

© ENI Editions - All rigths reserved - Algeria Educ

```
    envoyerMonException("Probleme entre les traitements");
    echo "Traitement 2\n";
} catch (Exception $e) {
    echo "\nException recuperee : \n\t\t".$e;
}
```

Résultats de traitement d'exception classique

```
Traitement 1

Exception recuperee :
            exception 'Exception' with message 'Envoi exception pour
raison(Probleme entre les traitements)' in
C:\wamp\www\php\objet\gestionException.php:17
Stack trace:
#0 C:\wamp\www\php\objet\gestionException.php(26):
envoyerException('Probleme entre ...')
#1 {main}
```

Nous allons voir dans un second exemple comment personnaliser votre gestion des erreurs par l'utilisation d'une classe d'exception personnalisée et ayant un comportement spécifique à votre programme ou votre projet.

Exemple de création d'une exception spécifique

```
class MonException extends Exception {
    public function __toString() {
    $str="\nException [";
    $str.="\n\tClasse: ".__CLASS__ ;
    $str.="\n\tMessage: ".$this->getMessage();
    $str.="\n\tFichier: ".$this->getFile();
    $str.="(".$this->getLine().")";
    $str.="\n\tPile d'appel: \n".$this->getTraceAsString();
    $str.="\n]";
    return $str;
 }
}
```

Exemple de traitement d'exception spécifique

```
function envoyerMonException($raison) {
    throw new MonException("Envoi de mon exception pour
raison($raison)");
}
try {
    echo "\nTraitement 3\n";
    envoyerMonException("Probleme entre les traitements bis
repetita");
    echo "\nTraitement 4\n";
} catch (Exception $e) {
    echo "\nException recuperee : \n\t\t".$e;
}
```

Résultats de traitement d'exception spécifique

```
Traitement 3

Exception recuperee :

Exception [
    Classe: MonException
    Message: Envoi de mon exception pour raison(Probleme
entre les traitements bis repetita)
    Fichier: C:\wamp\www\php\objet\gestionException.php( 21 )
    Pile d'appel:
#0 C:\wamp\www\php\objet\gestionException.php(35):
```


© ENI Editions - All rigths reserved - Algeria Educ

```
envoyerMonException('Probleme entre ...')
#1 {main}
]
```

h. API de réflectivité sur les classes

L'API de réflectivité est un ensemble de classes cohérentes et incluses dans le cœur de PHP permettant l'analyse des fonctions, des classes, des extensions et des configurations incluses dans votre installation de PHP.

La manière la plus simple d'utiliser cette API est en ligne de commande :

Syntaxe de base

Syntaxe	Description
$php -rf nomDeLaFonction	La commande restitue l'ensemble des informations concernant l'interface d'appel de la fonction nomDeLaFonction.
$php -rc nomDeLaClasse	La commande restitue l'ensemble des informations concernant l'interface de la classe nomDeLaClasse.
$php -re nomDeLExtension	La commande restitue l'ensemble des informations concernant les classes et les fonctions mises à disposition par l'extension nomDeLExtension.
$php -ri nomDeLExtension	La commande restitue l'ensemble des informations de configuration de l'extension nomDeLExtension.

Exemple d'utilisation de l'introspection de fonction

```
$php --rf fopen
Function [ <internal:standard> function fopen ] {

  - Parameters [4] {
    Parameter #0 [ <required> $filename ]
    Parameter #1 [ <required> $mode ]
    Parameter #2 [ <optional> $use_include_path ]
    Parameter #3 [ <optional> $context ]
  }
}
```

Exemple d'utilisation de l'introspection de classe

```
$php --rc SimpleXMLElement
Class [ <internal:SimpleXML> <iterateable> class SimpleXMLElement
implements Traversable ] {

  - Constants [0] {
  }

  - Static properties [0] {
  }

  - Static methods [0] {
  }

  - Properties [0] {
  }

  - Methods [14] {
    Method [ <internal:SimpleXML, ctor> final public method
__construct ] {
    }
```

© ENI Editions - All rigths reserved - Algeria Educ

```
    Method [ <internal:SimpleXML> public method asXML ] {
    }

    Method [ <internal:SimpleXML> public method saveXML ] {
    }

    Method [ <internal:SimpleXML> public method xpath ] {
    }

    Method [ <internal:SimpleXML> public method
registerXPathNamespace ] {
    }

    Method [ <internal:SimpleXML> public method attributes ] {
    }

    Method [ <internal:SimpleXML> public method children ] {
    }

    Method [ <internal:SimpleXML> public method getNamespaces ] {
    }

    Method [ <internal:SimpleXML> public method getDocNamespaces ] {
    }

    Method [ <internal:SimpleXML> public method getName ] {
    }

    Method [ <internal:SimpleXML> public method addChild ] {
    }

    Method [ <internal:SimpleXML> public method addAttribute ] {
    }

    Method [ <internal:SimpleXML> public method __toString ] {
    }

    Method [ <internal:SimpleXML> public method count ] {
    }
  }
}
```

Exemple d'utilisation de l'introspection d'extension

```
$php --re hash
Extension [ <persistent> extension #10 hash version 1.0 ] {

  - Constants [1] {
    Constant [ integer HASH_HMAC ] { 1 }
  }

  - Functions {
    Function [ <internal:hash> function hash ] {

      - Parameters [3] {
        Parameter #0 [ <required> $algo ]
        Parameter #1 [ <required> $data ]
        Parameter #2 [ <optional> $raw_output ]
      }
    }
    Function [ <internal:hash> function hash_file ] {

      - Parameters [3] {
        Parameter #0 [ <required> $algo ]
        Parameter #1 [ <required> $filename ]
        Parameter #2 [ <optional> $raw_output ]
      }
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    }
    Function [ <internal:hash> function hash_hmac ] {

      - Parameters [4] {
        Parameter #0 [ <required> $algo ]
        Parameter #1 [ <required> $data ]
        Parameter #2 [ <required> $key ]
        Parameter #3 [ <optional> $raw_output ]
      }
    }
    Function [ <internal:hash> function hash_hmac_file ] {

      - Parameters [4] {
        Parameter #0 [ <required> $algo ]
        Parameter #1 [ <required> $filename ]
        Parameter #2 [ <required> $key ]
        Parameter #3 [ <optional> $raw_output ]
      }
    }
    Function [ <internal:hash> function hash_init ] {

      - Parameters [3] {
        Parameter #0 [ <required> $algo ]
        Parameter #1 [ <optional> $options ]
        Parameter #2 [ <optional> $key ]
      }
    }
    Function [ <internal:hash> function hash_update ] {

      - Parameters [2] {
        Parameter #0 [ <required> $context ]
        Parameter #1 [ <required> $data ]
      }
    }
    Function [ <internal:hash> function hash_update_stream ] {

      - Parameters [3] {
        Parameter #0 [ <required> $context ]
        Parameter #1 [ <required> $handle ]
        Parameter #2 [ <optional> $length ]
      }
    }
    Function [ <internal:hash> function hash_update_file ] {

      - Parameters [3] {
        Parameter #0 [ <required> $context ]
        Parameter #1 [ <required> $filename ]
        Parameter #2 [ <optional> $context ]
      }
    }
    Function [ <internal:hash> function hash_final ] {

      - Parameters [2] {
        Parameter #0 [ <required> $context ]
        Parameter #1 [ <optional> $raw_output ]
      }
    }
    Function [ <internal:hash> function hash_copy ] {

      - Parameters [1] {
        Parameter #0 [ <required> $context ]
      }
    }
    Function [ <internal:hash> function hash_algos ] {

      - Parameters [0] {
      }
    }
  }
```

© ENI Editions - All rigths reserved - Algeria Educ

```
}
```

Exemple d'utilisation de l'introspection de configuration d'extension

```
$php --ri hash

hash

hash support => enabled
Hashing Engines => md2 md4 md5 sha1 sha224 sha256 sha384 sha512 ripemd128
ripemd160 ripemd256 ripemd320 whirlpool tiger128,3 tiger160,
3 tiger192,3 tiger128,4 tiger160,4 tiger192,4 snefru
snefru256 gost adler32 crc32 crc32b
salsa10 salsa20 haval128,3 haval160,3 haval192,3 haval224,
3 haval256,3 haval128,4 haval160,4 haval192,4 haval224,
4 haval256,4 haval128,5 haval160,5 haval192,5 haval224,5 haval256,5
```

3. Gestion de la bibliothèque Standard PHP

Dans la version 5.0 de PHP apparaît une nouvelle classe dans une extension appelée SPL comme Standard Php Librairie. Il s'agit d'une bibliothèque contenant une implémentation de la plupart des algorithmes fondamentaux tels que les listes, les arbres, les tables de hachage…

Dans cette version, une structure permettant d'indexer n'importe quel contenu par un objet quelconque apparaît : SplObjectStorage.

Le développeur PHP n'est plus limité à stocker dans un tableau des valeurs indexées par clé, caractères ou chaînes, il a maintenant la possibilité de le réaliser en utilisant n'importe quel objet.

Voici d'ailleurs l'erreur produite dans le cas de l'utilisation d'un objet comme clé de tableau :

```
< ?php
$arr=array();
$p1=new Profil('toto');
$arr[$p1]=array(1,2,3);
print_r($arr);
?>
```

```
Warning: Illegal offset type in arrayIndexee.php on line 4
```

a. La classe Profil de notre exemple

```
<?php

class Profil {
    private $pseudo;

    public function __construct($p='anonyme') {
        echo "\n* Constructeur de la classe: ".__CLASS__ .
" avec $p";
        $this->pseudo=$p;
    }
    public function __destruct() {
        echo "\n* Destructeur de la classe: ".__CLASS__;
        echo "\nManipulation de : ". $this;
        echo "\n";
    }
    public function __toString() {
           $chaine=__CLASS__;
           $chaine.= " [";
           $chaine.= "\n\t Pseudo: ";
           $chaine.= $this->pseudo;
           $chaine.= "\n]\n";
           return $chaine;
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    }
}
?>
```

b. L'objet SplObjetStorage en tant qu'ensemble

```
<?php

function __autoload($class_name)
{
    require_once "$class_name.class.php";
}

$parrainage=new SplObjectStorage();
for ($i=1;$i<3;$i++) {
    $p=new Profil("profil_$i");
    $parrainage->attach($p);
}

echo "\n\nNb element: ".count($parrainage)."\n";
$parrainage->rewind();
while($parrainage->valid()) {
    $index  = $parrainage->key();
    $object = $parrainage->current();

    echo "#######################################\n";
    echo "Index: $index\n";
    echo "Valeur: $object\n";
    $parrainage->next();
}

echo "\n#######################################\n";
print_r($parrainage);
echo "#######################################\n";
?>
```

L'utilisation d'un objet SplObjetStorage comme tableau à index est donc plus complexe, moins simple d'un simple tableau (array) PHP et donne peu de valeur ajoutée.

```
Nb element: 2
#######################################
Index: 0
Valeur: Profil [
         Pseudo: profil_1
]

#######################################
Index: 1
Valeur: Profil [
         Pseudo: profil_2
]


#######################################
SplObjectStorage Object
(
    [storage:SplObjectStorage:private] => Array
        (
            [000000007f43ba0400000000485d0534] => Array
                (
                    [obj] => Profil Object
                        (
                            [pseudo:Profil:private] => profil_1
                        )

                    [inf] =>
                )
```

© ENI Editions - All rigths reserved - Algeria Educ

```
            [000000007f43ba0500000000485d0534] => Array
                (
                    [obj] => Profil Object
                        (
                            [pseudo:Profil:private] => profil_2
                        )

                    [inf] =>
                )

        )

)
#####################################
```

c. L'objet SplObjetStorage en tant que tableau associatif

```
<?php

function __autoload($class_name)
{
    require_once "$class_name.class.php";
}

$parrainage=new SplObjectStorage();
$p2=$p=new Profil("profil_0");
for ($i=1;$i<3;$i++) {
    $p=$p2;
    $p2=new Profil("profil_$i");
    $parrainage[$p]=$p2;
}


echo "\n\nNb element: ".count($parrainage)."\n";
echo "Recherche avec index en objet: \n";
print_r($parrainage[$p]);

echo "Parcours complet des parrainages: \n";

$parrainage->rewind();
while($parrainage->valid()) {
    $index  = $parrainage->key();
    $object = $parrainage->current();
    $data   = $parrainage->getInfo();
    echo "#####################################\n";
    echo "Index: $index\n";
    echo "Valeur: $object\n";
    echo "Info: $data\n";
    $parrainage->next();
}

echo "\n#####################################\n";
print_r($parrainage);
echo "#####################################\n";
?>
```

L'utilisation d'objet comme clé d'un tableau associatif devient possible. Cependant, l'utilisation d'un parcours intégral de la structure n'est plus aussi simple que le parcours d'un tableau PHP avec la directive de base : foreach.

```
Nb element: 2
Recherche avec index en objet:
Profil Object
(
    [pseudo:Profil:private] => profil_2
)
Parcours complet des parrainages:
#####################################
```


© ENI Editions - All rigths reserved - Algeria Educ

```
Index: 0
Cle: Profil [
         Pseudo: Parrain
]

Valeur: Profil [
         Pseudo: Parrainnee-1
]

#####################################
Index: 1
Cle: Profil [
         Pseudo: Parrainnee-1
]

Valeur: Profil [
         Pseudo: Parrainnee-2
]


#####################################
SplObjectStorage Object
(
    [storage:SplObjectStorage:private] => Array
        (
            [0000000056dc358f000000006f4630fe] => Array
                (
                    [obj] => Profil Object
                        (
                            [pseudo:Profil:private] => Parrain
                        )

                    [inf] => Profil Object
                        (
                            [pseudo:Profil:private] => Parrainnee-1
                        )

                )

            [0000000056dc358e000000006f4630fe] => Array
                (
                    [obj] => Profil Object
                        (
                            [pseudo:Profil:private] => Parrainnee-1
                        )

                    [inf] => Profil Object
                        (
                            [pseudo:Profil:private] => Parrainnee-2
                        )

                )

        )

)
#####################################
```

4. La gestion des données

La gestion des données est grandement améliorée dans cette version, permettant aux développeurs de manipuler plus facilement les données dans les deux formes les plus classiques actuelles : le SQL et le XML.

Il faut noter que SQLite, la base au format fichier, devient une extension par défaut de PHP avec une API de base très simple à utiliser.

La couche d'abstraction très chère à chaque langage de programmation afin de faire valoir son indépendance vis-à-vis de la programmation autour de bases de données fait aussi son apparition dans le langage PHP : PDO.

© ENI Editions - All rigths reserved - Algeria Educ

L'acronyme PDO signifie *Portable Data Object* ou plus littéralement Objet de Données Portable. Il s'agit d'objet PHP permettant de manipuler les bases de données de manière uniforme et de réduire le portage d'une application vers une autre base de données en un minimum de temps.

L'avantage de PDO est qu'il met fin à une guerre du genre entre de nombreux projets en PHP permettant d'ajouter cette fonctionnalité manquante dans les versions précédentes.

a. Gestion du XML via SimpleXML

Cette API offre un objet de manipulation de données XML et deux fonctions permettant respectivement de lire des données XML depuis un fichier ou depuis une chaîne de caractères.

```
<?php
libxml_use_internal_errors(true);
$sxe = simplexml_load_string("<?xml version='1.0'><casse>
</casse><ouvert><ferme>");
if (!$sxe) {
    echo "Erreur de chargement de la chaine XML.\n";
    foreach(libxml_get_errors() as $error) {
        echo "\t", $error->message;
    }
}
?>
```

```
Erreur de chargement de la chaine XML.
        Blank needed here
        parsing XML declaration: '?>' expected
        Extra content at the end of the document
```

Version avec lecture d'une chaîne de caractères

```
<?php
$xmlstr = <<<XML
<?xml version='1.0' standalone='yes'?>
<contacts>
 <contact>
  <nom>Renouard</nom>
  <prenom>Jean-Marie</prenom>
  <adresse>8, boulevard Clemenceau</adresse>
  <codePostale>75003</codePostale>
  <ville>Paris</ville>
 </contact>
  <contact>
  <nom>Picard</nom>
  <prenom>Claude</prenom>
  <adresse>14, avenue des allouettes</adresse>
  <codePostale>13000</codePostale>
  <ville>Marseille</ville>
 </contact>
   <contact>
  <nom>Robert</nom>
  <prenom>Pierre</prenom>
  <adresse>1, rue Saint Victoire</adresse>
  <codePostale>78000</codePostale>
  <ville>Versailles</ville>
 </contact>
</contacts>
XML;

$xml = simplexml_load_string($xmlstr);
foreach ($xml->contact as $contact) {
   echo "\n* ". $contact->prenom." ".$contact->nom;
   echo "\n\t". $contact->adresse."\n\t".$contact->codePostale."
".$contact->ville;
}
echo "\n";
```


© ENI Editions - All rigths reserved - Algeria Educ

```
?>
```

Version avec lecture d'un fichier XML

```
<?php
$xml = simplexml_load_file("data.xml");
foreach ($xml->contact as $contact) {
   echo "\n* ". $contact->prenom." ".$contact->nom;
   echo "\n\t". $contact->adresse."\n\t".$contact->codePostale."
".$contact->ville;
}
echo "\n";
?>
```

Résultat

```
* Jean-Marie Renouard
        8, boulevard Clemenceau
        75003 Paris
* Claude Picard
        14, avenue des allouettes
        13000 Marseille
* Pierre Robert
        1, rue Saint Victoire
        78000 Versailles
```

b. Gestion de bases SQLite

Un exemple d'utilisation de l'API de base de SQLite dans PHP.

Son avantage, encore une fois, est la simplicité et la rapidité de mise en œuvre.

```
<?php



$base='/tmp/base3.db';
$sql="select * from T1";

unlink ($base);

$db = sqlite_open($base, 0777, $sqliteerreur);
if (!$db) {
    die($sqliteerreur);
}

sqlite_query($db, 'CREATE TABLE T2 (id varchar(10))');

sqlite_query($db, "INSERT INTO T2 VALUES ('fnord')");
sqlite_query($db, "INSERT INTO T2 VALUES ('fnord2')");
sqlite_query($db, "INSERT INTO T2 VALUES ('fnord3')");

echo "\nRecuperation 1 ligne: ";
$result = sqlite_query($db, 'select id from T2');
var_dump(sqlite_fetch_array($result));

echo"\nRecuperation toutes les lignes: ";
$result2 = sqlite_query($db, 'select id from T2');
var_dump(sqlite_fetch_all($result2, SQLITE_ASSOC ));
?>
```

```
Recuperation 1 ligne: array(2) {
  [0]=>
  string(5) "fnord"
  ["id"]=>
```

© ENI Editions - All rigths reserved - Algeria Educ

```
  string(5) "fnord"
}

Recuperation toutes les lignes: array(3) {
  [0]=>
  array(1) {
    ["id"]=>
    string(5) "fnord"
  }
  [1]=>
  array(1) {
    ["id"]=>
    string(6) "fnord2"
  }
  [2]=>
  array(1) {
    ["id"]=>
    string(6) "fnord3"
  }
}
```

c. Gestion des bases de données via PDO

Un exemple de portage de l'exemple précédent en utilisant l'API PDO.

```
<?php

$dsn='sqlite:/tmp/base.db';
$sql="select * from T1";

$dbh=NULL;
try {
    $dbh=new PDO($dsn);
} catch(PDOException $e){
    echo "Erreur de connexion vers $dsn: ".$e->getMessage();
    exit(1);
}

$sth = $dbh->prepare($sql);
$sth->execute();
if ($sth->errorCode()=== '00000') {
        $result = $sth->fetchAll();
} else {
    print_r($sth->errorInfo());
}

$sth->closeCursor();

print_r($result);
?>
```

Mise en place de fonctions utilitaires pour PDO

```
<?php

function getDbConnection($dsn, $user, $pass) {
        $ret=null;
        try{
                $ret=new PDO($dsn, $user, $pass);
        }
        catch(PDOException $e){
                echo "Erreur de connexion vers $dsn: ".$e->getMessage();
                $ret=null;
                throw new Exception($e);
        }
        print_r($ret->errorInfo());
```


© ENI Editions - All rigths reserved - Algeria Educ

```
        if ($ret==NULL) throw new Exception("NULL connection");

        return $ret;
}


function getNbEnregistrements($dbh, $tblName) {
        $nb=0;
        try  {
                $sth = $dbh->prepare('SELECT COUNT(*) FROM '.$tblName);
                $sth->execute();
                $results=$sth->fetchAll(PDO::FETCH_COLUMN, 0);
                $nb=$results[0];
        } catch (Exception $e) {
                echo "$e";
        }
        $sth->closeCursor();
        return $nb;
}

function getSqlResultat($dhn, $sql, $mode=PDO::FETCH_BOTH) {
    $sth = $dbh->prepare($sql);
    $sth->execute();
    if ($sth->errorCode()=== '00000') {
            $result = $sth->fetchAll($mode==NULL?:$mode);
    } else {
        print_r($sth->errorInfo());
    }

    $sth->closeCursor();
    return $result;
}

function getSqlAssocResultat($dhn, $sql) {
    return getSqlResultat($dhn, $sql, PDO::FETCH_ASSOC);
}

function getSqlObjetResultat($dhn, $sql) {
    return getSqlResultat($dhn, $sql, PDO::FETCH_OBJ);
}

?>
```

5. Les évolutions entre PHP 5 et PHP 5.3

Version	Date	Faits marquants de la version
5.0.0	13 juillet 2004	Nouveau moteur Zend2 avec la nouvelle syntaxe objet.
5.1.0	24 novembre 2005	Amélioration des performances avec l'introduction de compilateur de variables dans le moteur PHP.
5.2.0	2 novembre 2006	Activation des filtres validation et nettoyage par défaut.
5.2.11	16 septembre 2009	Version de mises à jour et de correctifs de sécurité.
5.2.12	17 décembre 2009	Version de mises à jour et de correctifs de sécurité.

© ENI Editions - All rigths reserved - Algeria Educ

Les points clés de la version 5.3

La version 5.3 est sortie le 30 juin 2009, soit quatre ans après la première version de la version 5. La dernière version corrective date du 19 novembre 2009 et concerne des corrections de bugs et patches de sécurité.

L'évolution de PHP entraîne aussi un certain nombre de sorties de code vers les dépôts d'extension C (PECL) ou les dépôts de classes PHP (PEAR).

L'ensemble des fonctions de l'extension ereg_* sont sorties du noyau du langage et sont relayées vers le dépôt PECL.

Mais au fond, qu'apporte cette version 5.3 par rapport aux versions précédentes en dehors des classiques corrections de bugs et de failles de sécurité ?

La version 5.3 de PHP est la première, de par les évolutions telles que l'amélioration des performances et l'amélioration de la mémoire consommée, à permettre enfin d'utiliser des scripts PHP comme traitements « batchs » capables de libérer la mémoire utilisée.

En effet, PHP en tant que langage pour le Web est adapté à des exécutions rapides et à cycle de vie court (le temps de la génération et de l'envoi du résultat de la page).

Avec ces améliorations, dont l'introduction d'un ramasse-miettes, PHP s'offre enfin la possibilité de devenir un langage de programmation à part entière et de sortir du milieu Web auquel il a été trop longtemps cantonné.

1. Apparition des espaces de nommage

Les espaces de nommage ont été souvent montrés du doigt par les communautés de développeurs comme une fonctionnalité des langages de programmation manquante dans le PHP.

Sans espace de nommage, PHP a poussé l'ensemble des fonctions et des classes du cœur et les extensions dans le même espace de nommage.

Ceci est problématique car il ne permet pas d'utiliser des classes ou des fonctions similaires selon l'espace de nommage utilisé. Cette lacune a entraîné une multiplication du nommage des fonctions et des classes. Les approches pour contourner ce point n'ont, hélas, pas abouti sur l'utilisation d'une norme de nommage des fonctions et des classes uniforme pour l'ensemble des éléments.

a. Qu'est-ce qu'un espace de nommage ?

Un espace de nommage est une zone spécifique permettant de regrouper des fonctions et des classes autour d'une même thématique ou technologie.

La terminologie des éléments peut être entièrement mise en relation avec l'espace de nommage et permet de séparer et de distinguer l'implémentation de deux éléments de nom similaire dans votre code en fonction de l'espace de nommage utilisé.

b. Syntaxe des espaces de nommage

Syntaxe de déclaration

Pour déclarer un espace de nommage, rien de plus simple : il suffit d'utiliser la syntaxe suivante, juste avant de définir les classes et les fonctions de votre espace.

```
namespace nomDeLEspaceDeNommage ;
```

La seule chose qu'il est important de noter est que votre espace de nommage doit être le premier élément du script.

Il est possible de définir plusieurs espaces de nommage dans un même fichier. Cependant, la déclaration du premier espace de nommage doit être réalisée en premier.

Syntaxe de déclaration d'un sous-espace de nommage

Pour déclarer un sous-espace de nommage ou une hiérarchie d'espaces de nommage, il suffit d'utiliser la syntaxe suivante juste avant de définir les classes et les fonctions de votre espace. Le caractère \ sera utilisé comme

© ENI Editions - All rigths reserved - Algeria Educ

séparateur entre le sous-espace et l'espace de nommage parent.

```
namespace nomDeLEspaceDeNommagePrincipal\
nomDeLEspaceDeNommageSecondaire\ nomDeLEspaceDeNommageTertiaire ;
```

La seule chose qu'il est important de noter est que vos espaces de nommage doivent être séparés par le caractère \.

c. Exemples de déclaration d'espace de nommage

Exemple de déclaration d'un espace de nommage

Nous allons déclarer un espace de nommage simple avec une variable et une fonction.

```
< ?php
namespace maZoneProjet ;
$VERSION=0.001 ;
function calculerLeCarre($i) {
    return $i*$i ;
}
?>
```

Exemple de déclaration d'espaces de nommage multiples

Nous allons déclarer plusieurs espaces de nommage avec, pour chacun, une variable et une fonction.

```
< ?php
namespace maZoneProjet ;
$VERSION=0.001 ;
function calculerLeCarre($i) {
    return $i*$i ;
}

namespace maZoneProduit ;
$VERSION=1.0 ;
function calculerProfit($i) {
    return \maZoneProjet\calculerLeCarre($i);
}
?>
```

Exemple de déclaration de sous-espaces de nommage

Nous allons déclarer plusieurs espaces de nommage avec, pour chacun, une variable et une fonction.

```
< ?php
namespace monEspacePrincipal ;
function afficher() {
    echo « Principal ». __NAMESPACE__ ;
}

namespace monEspacePrincipal\monEspaceSecondaire;
function afficher() {
    echo « Secondaire : ». __NAMESPACE__ ;
}
?>
```

d. Utilisation des espaces de nommage

Les espaces de nommage s'utilisent de manière à différencier les éléments utilisés dans votre code en fonction des différents espaces de nommage disponibles.

© ENI Editions - All rigths reserved - Algeria Educ

Il existe cinq techniques conventionnelles pour utiliser les espaces de nommage :

- Définir votre code d'utilisation dans le même espace de nommage.
- Utiliser le nom absolu de l'espace de nommage à utiliser.
- Utiliser le nom relatif de l'espace de nommage à utiliser.
- Importer les éléments d'un espace de nommage dans votre espace.
- Créer des alias d'élément dans votre espace courant.

e. Syntaxe d'utilisation des espaces de nommage

Syntaxe d'utilisation d'un espace de nommage

```
namespace nomDeLEspaceDeNommagePrincipal ;
```

Tous les éléments déclarés dans l'espace de nommage nomDeLEspaceDeNommagePrincipal sont accessibles comme les éléments de l'espace de nommage global. Attention donc aux conflits entre les éléments de l'espace global et ceux de l'espace nomDeLEspaceDeNommagePrincipal.

Exemple d'utilisation par nommage absolu

```
\monEspacePrincipal\monEspaceSecondaire\afficher()
```

Tous les éléments nommés de manière absolue commencent par le caractère \.

Exemple d'utilisation par nommage relatif

```
monEspaceSecondaire\afficher()
```

Il est impératif que votre code soit écrit ou utilise déjà l'espace de nommage parent afin que vous puissiez atteindre de manière relative le sous-espace de nommage.

Syntaxe d'importation d'élément d'un espace de nommage

```
use nomDeLEspaceDeNommagePrincipal\element ;
use \nomDeLEspaceDeNommagePrincipal\element ;
```

Il est important de bien noter que l'import n'est réalisé que pour chaque élément et ne réalise pas une importation massive des éléments d'un espace de nommage comme avec le mot clé : `namespace`.

Syntaxe de création d'alias d'un espace de nommage

```
use nomDeLEspaceDeNommagePrincipal\element as MonElement;
use \nomDeLEspaceDeNommagePrincipal\element as MonElement;
```

Il s'agit d'un mécanisme permettant de créer un alias dans votre code afin de simplifier la syntaxe de base. Attention, à ne pas entraîner des conflits ou des ambiguïtés en choisissant le nom de vos alias.

© ENI Editions - All rigths reserved - Algeria Educ

2. L'introduction de la liaison tardive

L'opérateur `self` permet de brancher le code sur les informations de la méthode de la classe qui contient l'appel à cet opérateur.

Il n'est donc pas possible d'obtenir un comportement dynamique au travers des données statiques.

La liaison tardive des appels de méthode sur une classe permet de choisir toujours la méthode la plus spécifique contenue dans la classe la plus dérivée de la hiérarchie de l'héritage.

La version 5.3.0 apporte la possibilité de profiter de cette fonctionnalité pour l'ensemble des données statiques. Cette fonctionnalité s'appelle la liaison tardive statique.

Pour la mettre en place, il suffit d'utiliser l'opérateur `static` en lieu et place de l'opérateur `self`.

a. Syntaxe des cinq opérateurs et variable de portée

Nom	Exemple	Description
$this	$this->nom	La variable this permet d'accéder aux propriétés de l'objet à l'intérieur de celui-ci.
parent::	parent::__construct()	L'opérateur parent permet d'invoquer explicitement une méthode de la classe mère.
static::	static::jouer()	L'opérateur static permet d'invoquer une méthode de manière dynamique, c'est-à-dire en cours d'exécution. L'opérateur ne renvoie plus la classe dans laquelle l'appel est implémenté mais dans la classe de l'objet utilisé lors de l'appel.
self::	self::controle()	L'opérateur self permet d'invoquer une méthode de la classe dans laquelle l'appel est implémenté.
NomDeCLasse ::	Mere ::Saluer()	Le nom de la classe peut être utilisé comme opérateur de portée afin de définir explicitement la classe contenant la propriété.

b. Exemples d'utilisation de self et static

Exemple d'utilisation du mot clé self

```
<?php

class Mere {
    public static function jesuis() {
          return " JeSuis [". __CLASS__. "]";
    }

    public static function jelance() {
          echo " JeLance [ ". self::jesuis(). " ]";
    }
}

class Fille extends Mere {
    public static function jesuis() {
          return " JeSuis [". __CLASS__. "]";
    }
}

echo Fille::jelance();
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

Résultats de l'utilisation du mot clé self

```
JeLance [ JeSuis [Mere] ]
```

Exemple d'utilisation du mot clé static

```
<?php

class Mere {
    public static function jesuis() {
           return " JeSuis [". __CLASS__. "]";
    }

    public static function jelance() {
        echo " JeLance [ ". self::jesuis(). " ]";
    }
}

class Fille extends Mere {
    public static function jesuis() {
           return " JeSuis [". __CLASS__. "]";
    }
}

echo Fille::jelance();
?>
```

Résultats de l'utilisation du mot clé static

```
JeLance [ JeSuis [FilleStatic] ]
```

3. Apparition des labels et des sauts de code

Les labels et les sauts sont des astuces de programmation permettant d'alléger la syntaxe.

Les sauts introduisent la notion de rupture de code décriée par une grande partie des développeurs.

En effet, un code sans saut possède une forme linéaire

1. appel1() ;
2. appel2() ;
3. Tant que condition Alors
   - appel3()
   - Fin Tant que

Le saut casse clairement la logique procédurale d'un code ; il faut donc éviter de tomber dans des points de vue extrêmes. En effet, un saut permet souvent de simplifier grandement du code en sautant des blocs de code.

L'alternative consiste à charger le code de test de condition if dans l'ensemble du code et d'augmenter la complexité cyclomatique, c'est-à-dire le nombre de conditions et de boucles imbriquées dans chaque fonction.

Pas de label possible à l'intérieur d'une boucle car les variables de parcours ne sont pas forcément correctement initialisées, comme par exemple la première clause d'une boucle for.

a. Syntaxe de base d'un saut et d'un label

Syntaxe du label

© ENI Editions - All rigths reserved - Algeria Educ

```
nomDeLabel :
```

Syntaxe du renvoi de label

```
goto nomDeLabel ;
```

b. Exemple d'utilisation des sauts et des labels

```
<?php

$tab=array(9, 8, 7, 6, 5, 4, 3, 2, 1, 0);
$i=0;
debut:
echo "\n* \$tab[$i]=".$tab[$i];
$i++;
if ($i<count($tab)) goto debut;
echo "\nFin de boucle pour";

?>
```

Ce code est une réécriture d'une boucle for sans opérateur for ni opérateur while.

Le code paraît ici plus linéaire ; en fait il n'en est rien !

En effet, l'utilisation de labels et de sauts engendre le sentiment d'un code plus simple alors qu'en réalité, il peut poser de véritables problèmes dans le suivi d'exécution et de mise au point des programmes.

4. Les fonctions magiques pour les appels statiques

La fonction __invoke() n'apparaît qu'en version 5.3 et permet à un objet d'une classe d'être utilisé comme une fonction.

Il est à noter cependant qu'il faut connaître le nombre d'arguments au préalable, il n'est donc pas possible de recevoir des appels avec des paramètres de longueur variable. Il faut donc ruser en lui passant, par exemple, un tableau PHP.

a. Syntaxe de base

Description de la méthode magique __invoke :

Méthodes de classe	Description
__invoke($params)	Méthode d'interception des appels utilisant l'objet comme une fonction. La méthode __invoke() prend 1 paramètre : le paramètre d'appel.

b. Exemple d'utilisation

Exemple d'utilisation de la méthode magique __invoke()

```
<?php

class MonTruc {
    public function __invoke($p, $p2="p2") {
            echo __CLASS__.": Je vais me comporter comme une
fonction avec paramètre [ ".print_r($p, true)., ".print_r($p2,
true)." ]\n";
    }
}

$mt=new MonTruc();
```


© ENI Editions - All rigths reserved - Algeria Educ

```
$mt("toto");
$mt("toto2", 'rr', 5);
$mt(array("cool", "toto2", "rr", "5"));
?>
```

5. L'apparition de NOWDOC et HEREDOC

Le Heredoc et le Nowdoc sont des moyens de stocker dans une variable du contenu chaîne de caractères. La valeur se trouve donc sur plusieurs lignes.

```
<?php
$nom="Tata";
$age=94;

$maValeur= <<<EOF
Bonjour $nom,

Je te souhaite un joyeux ${age}ème anniversaire.

Courage,
Ton neveu favori.
EOF;

echo $maValeur;
?>
```

La particularité d'un Heredoc est que le contenu est analysé et le nom des variables est substitué par leur valeur.

Le Heredoc en version 5.3 permet en plus d'initialiser :

- Les attributs statiques dans les classes
- Les attributs d'objet dans les classes
- Les valeurs d'un tableau PHP

a. Syntaxe de base d'un Heredoc

Le Heredoc comprend deux formes :

Exemple de la première forme

```
$val <<<LABEL
contenu
sur
plusieurs
lignes
.
LABEL;
```

Exemple de la deuxième forme

```
$val <<<"LABEL"
contenu
sur
plusieurs
lignes
.
une Variable :$nom
LABEL;
```

© ENI Editions - All rigths reserved - Algeria Educ

b. Spécificités d'un Nowdoc

Le Heredoc permet, nous l'avons vu, d'interpréter les variables contenues dans la chaîne comme lors d'une initialisation du type chaîne entre guillemets.

```
$phrase= « Bonjour, $nom » ;
```

Pour remédier à ce problème, PHP 5.3 introduit un nouvel élément ayant un comportement similaire sans analyser ni substituer le contenu des données.

Son comportement va être similaire à :

```
$phrase= ' Bonjour, $nom ' ;
```

Les variables contenues à l'intérieur ne seront pas traduites ni substituées par leur valeur.

c. Syntaxe de base d'un Nowdoc

```
$val <<<'LABEL'
contenu
sur
plusieurs
lignes
.
une variable : $nom
LABEL;
```

6. Support des fonctions lambda

Les fonctions anonymes apparaissent comme une évolution naturelle du langage afin de supporter cette fonctionnalité présente aujourd'hui dans de nombreux langages de script.

Les fonctions anonymes permettent de déclarer des fonctions sans nom.

a. Syntaxe de base

Définition d'une fonction anonyme

```
$uneFonctionAnonyme = function ( paramètres ...) { code de la fonction } ;
```

La fonction peut être stockée dans une variable simple ou dans un élément de tableau car ce dernier est aussi, par définition, une variable.

Utilisation d'une fonction anonyme

```
$uneFonctionAnonyme() ;
$uneFonctionAnonyme[indice]() ;
$uneFonctionAnonyme( paramètres...) ;
$uneFonctionAnonyme[indice]( paramètres...) ;
```

b. Exemple de fonctionnement

```
<?php
$tab=array();

$tab[0]=function() { echo "fonction 0";};
$tab[1]=function() { echo "fonction 1";};
$tab[2]=function() { echo "fonction 2";};
```


© ENI Editions - All rigths reserved - Algeria Educ

```
$tab[3]=function() { echo "fonction 3";};
$tab[4]=function() { echo "fonction 4";};
for ($i=0; $i<count($tab); $i++) {
    echo "\n* Appel de la fonction de \$tab[$i] :";
    $tab[$i]();
}
?>
```

Résultats d'exécution

```
$ php anon1.php

* Appel de la fonction de $tab[0] :fonction 0
* Appel de la fonction de $tab[1] :fonction 1
* Appel de la fonction de $tab[2] :fonction 2
* Appel de la fonction de $tab[3] :fonction 3
* Appel de la fonction de $tab[4] :fonction 4
```

7. Support natif des archives PHAR

L'archive PHAR est un moyen utile de construire une application tout-en-un pour PHP. En effet, le support de PHAR existe, cependant il intègre nativement PHP depuis cette dernière version.

L'avantage du format PHAR est qu'il standardise le format d'échange d'archives de code. Il apporte de nombreuses fonctionnalités comme la signature d'archive, la possibilité d'auto exécution et bien d'autres encore. Parmi les avantages les plus significatifs, PHAR permet de créer directement en PHP et surtout d'utiliser directement une archive comme un fichier classique.

a. Exemple de création d'une archive PHAR

```
<?php

$phar_file=$argv[1];

if (file_exists($phar_file)) {
    echo "\n* Removing old $phar_file";
    unlink($phar_file);
}
echo "\n* Creating $phar_file file.";
$phar = new Phar($phar_file);
$phar->compressAllFilesGZ();

print_r($phar);
for ($i=2; $i<count($argv); $i++) {
    echo "\n* adding $phar_file <= ".$argv[$i];
    $phar->addFile($argv[$i]);
}
?>
```

8. Introduction d'un ramasse-miettes

Le ramasse-miettes est une fonctionnalité de la plupart des langages de script, dont le rôle principal est de rechercher les zones mémoire qui ne sont plus accessibles directement et de libérer cette mémoire. Les langages compilés (tels que le langage C ou C++) n'ont pas de mécanisme de ramasse-miettes et donc, la responsabilité de la libération de la mémoire repose sur les épaules des développeurs.

Les pertes de mémoire ne sont pas importantes lors d'un fonctionnement très court tel que la génération de pages web. En effet, la mémoire est récupérée intégralement par le système à la fin de l'exécution du script.

Les problèmes arrivent quand on utilise PHP pour des traitements de masse nécessitant beaucoup de mémoire au travers de variables, tableaux et objets.

Sans ramasse-miettes, la mémoire utilisée grandit au fur et à mesure de l'avancement du traitement. Aucune parade n'existe et, sans ramasse-miettes, pas de possibilité de garantir le fonctionnement avec l'augmentation du nombre

© ENI Editions - All rigths reserved - Algeria Educ

de données traitées. Le script PHP aura, comme limite mémoire, celle déclarée dans la directive allow_memory ou bien l'ensemble de l'espace RAM et de l'espace Swap de votre système d'exploitation. Cette dernière possibilité est une catastrophe car votre script viendra utiliser l'ensemble des ressources nécessaires aux autres programmes de votre système.

a. Exemple de script de traitement long

```
<?php

function __autoload($class_name)
{
    require_once "$class_name.class.php";
}

//gc_disable();
$baseMemory = memory_get_usage();
$basePeakMemory=memory_get_peak_usage();
$b=new Profil("b");
$c=new Profil("c");
for ( $i = 0; $i <= 500000; $i++ )
{
    $c=$b;
    $a=$b;
    $a = new Profil("$i");
    $c->self=$b;
    $b->self=$c;
    $a->self=$a;
    if ( $i % 10000 === 0 )
    {
        echo sprintf( '%8d %8d %8d', $i , (memory_get_usage() -
$baseMemory),(memory_get_peak_usage() - $basePeakMemory));
        echo "\n";
    }
    $a->setParrain($b);
    $b->setParrain($c);

}
?>
```

```
$ time php -dzend.enable_gc=0 -dmemory_limit=-1 gc1.php
       0     6144    11600
   10000  4177728  4172336
   20000  8382040  8376648
   30000 12062064 12056672
   40000 16790656 16785264
   50000 20470688 20465296
   60000 24150704 24145312
   70000 29927880 29922488
   80000 33607904 33602512
   90000 37287920 37282528
  100000 40967952 40962560
  110000 44647968 44642576
  120000 48327992 48322600
  130000 52008016 52002624
  140000 59882336 59876944
  150000 63562368 63556976
  160000 67242384 67236992
  170000 70922408 70917016
  180000 74602432 74597040
  190000 78282448 78277056
  200000 81962480 81957088
  210000 85642496 85637104
  220000 89322520 89317128
  230000 93002544 92997152
  240000 96682560 96677168
  250000 100362592 100357200
  260000 104042608 104037216
```


© ENI Editions - All rigths reserved - Algeria Educ

```
  270000 116111240 116105848
  280000 119791264 119785872
  290000 123471280 123465888
  300000 127151312 127145920
  310000 130831328 130825936
  320000 134511352 134505960
  330000 138191376 138185984
  340000 141871392 141866000
  350000 145551424 145546032
  360000 149231440 149226048
  370000 152911464 152906072
  380000 156591488 156586096
  390000 160271504 160266112
  400000 163951536 163946144
  410000 167631552 167626160
  420000 171311576 171306184
  430000 174991600 174986208
  440000 178671616 178666224
  450000 182351648 182346256
  460000 186031664 186026272
  470000 189711688 189706296
  480000 193391712 193386320
  490000 197071736 197066344
  500000 200751760 200746368

real    0m4.526s
user    0m0.015s
sys     0m0.000s
```

Avec la désactivation du ramasse-miettes, l'augmentation de la mémoire consommée est prononcée et le script PHP termine son exécution avec 200 Mo de mémoire utilisée.

Cela rend, de facto, impossible l'utilisation du PHP pour des traitements batch de large volume de données dans un entrepôt de données par exemple.

```
$ time php -dzend.enable_gc=1 -dmemory_limit=-1 gc1.php
       0     6144    11600
   10000   498424  4171496
   20000   498792  4171864
   30000   499160  4171864
   40000   499528  4171864
   50000   499896  4171864
   60000   500264  4171864
   70000   500632  4171864
   80000   501000  4171864
   90000   501368  4171864
  100000   501736  4171864
  110000   502104  4171896
  120000   502472  4171896
  130000   502840  4171896
  140000   503208  4171912
  150000   503576  4171912
  160000   503944  4171912
  170000   504312  4171912
  180000   504680  4171912
  190000   505048  4171912
  200000   505416  4171912
  210000   505784  4171920
  220000   506152  4171920
  230000   506520  4171920
  240000   506888  4171920
  250000   507256  4171936
  260000   507624  4171936
  270000   507992  4171936
  280000   508360  4171936
  290000   508728  4171944
  300000   509096  4171944
  310000   509464  4171968
  320000   509832  4171968
  330000   510200  4171968
```

© ENI Editions - All rigths reserved - Algeria Educ

```
  340000   510568  4171968
  350000   510936  4171968
  360000   511304  4171968
  370000   511672  4171968
  380000   512040  4171968
  390000   512408  4171976
  400000   512776  4171976
  410000   513144  4171976
  420000   513512  4171976
  430000   513880  4171976
  440000   514248  4171976
  450000   514616  4171992
  460000   514984  4171992
  470000   515352  4171992
  480000   515720  4171992
  490000   516088  4171992
  500000   516456  4171992

real    0m4.946s
user    0m0.000s
sys     0m0.015s
```

Le résultat est visible : dans le cas de l'utilisation du ramasse-miettes, la consommation mémoire reste stable dans le temps (autour de 4 Mo) et un léger temps d'exécution est présent, dû à l'exécution des libérations mémoire.

9. Ajout d'élément dans la SPL

Bien que déjà présente en version 5, la bibliothèque standard PHP (SPL) se trouve grandement enrichie au passage de cette version en intégrant la plupart des algorithmes classiques de base.

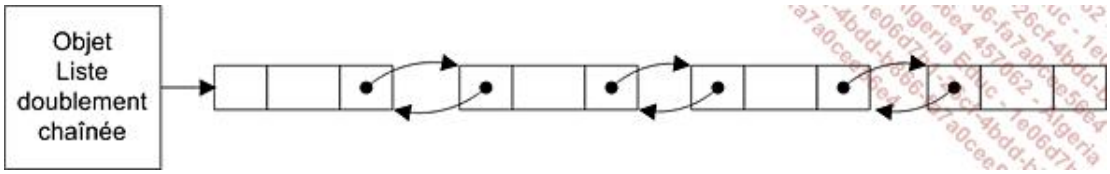
La bibliothèque standard PHP a pour objectif d'offrir aux développeurs PHP un ensemble d'algorithmes préétablis implémenté de manière native et maintenu au sein de PHP.

Parmi les algorithmes classiques, nous trouvons les listes chaînées, les files, les piles, les files à priorité, les arbres binaires et les tableaux à taille fixe.

Ces structures précodées sont présentes avec l'ensemble des fonctions permettant de les manipuler de la manière la plus efficace possible.

a. Les listes doublement chaînées

Les listes doublement chaînées sont issues d'une structure de données classique en algorithmique (la liste chaînée) à laquelle est ajouté un chaînage arrière.



Chaque nœud de données est capable de fournir un accès à l'élément suivant ou précédent, si cet élément existe.

De cette composition de base, il devient rapidement possible d'implémenter une pile de données ou une file.

En effet, la seule différence entre la file et la pile réside dans l'ordre de classement des éléments.

Dans le cas de la pile, le premier élément traité est le plus récemment inséré. L'image de la pile de papier est très intéressante comme métaphore, on empile des documents puis on dépile en commençant par le haut de la pile de documents.

Dans le cas de la queue ou file, l'analogie la plus simple est celle du guichet de l'administration. Celui qui est arrivé avant vous passe avant vous !

Les algorithmes à base de listes chaînées sont les suivants :

- SplDoublyLinkedList
- SplStack

© ENI Editions - All rigths reserved - Algeria Educ

- Splqueuee

b. Exemple d'utilisation de listes chaînées

Cet exemple met en évidence la possibilité d'utiliser la structure SplDoublyLinkedList comme une structure souple alliant les avantages d'une file et d'une pile.

```
<?php

function __autoload($class_name)
{
    require_once "$class_name.class.php";
}
$dll = new SplDoublyLinkedList();

// Ajoute à la fin
$dll->push (new Profil("A la fin 1"));
$dll->push (new Profil("A la fin 2"));

// Ajoute au début
$dll->unshift( new Profil("A debut 3"));

foreach ($dll as $profil)  {
 echo $profil."\n";
}
?>
```

Résultats d'exécution de script

```
$ php spldll1.php
Profil [
         Pseudo: A debut 3
]

Profil [
         Pseudo: A la fin 1
]

Profil [
         Pseudo: A la fin 2
]
```

c. Exemple d'utilisation de pile

```
<?php

function __autoload($class_name)
{
    require_once "$class_name.class.php";
}
$stack = new SplStack();

$stack[] = new Profil("1");
$stack[] = new Profil("2");
$stack[] = new Profil("3");

foreach ($stack as $profil)  {
 echo $profil."\n";
}
?>
```

Résultats d'exécution de script

© ENI Editions - All rigths reserved - Algeria Educ

```
$ php splstack1.php
Profil [
         Pseudo: 3
]

Profil [
         Pseudo: 2
]

Profil [
         Pseudo: 1
]
```

L'exécution met en évidence qu'en fonctionnement en mode pile, le dernier arrivé est aussi le premier à sortir de la structure. Cette forme de stockage correspond bien à des cas de fonctionnement type dossier administratif, c'est-à-dire que le dossier le plus urgent (ou important) est sorti et mis en haut de la pile.

d. Exemple d'utilisation de file

Dans cet exemple, le mode de retrait est le mode suppression.

Nous utilisons la file pour stocker des fonctions anonymes et, à chaque fois qu'elles sont traitées depuis la file, elles sont exécutées systématiquement puis réintroduites dans la file avec une probabilité de 50%.

```
<?php
$q = new Splqueuee();
$q->setIteratorMode(SplQueue::IT_MODE_DELETE);

// Ajout de Tâche dans la queue
$q[]=function() { echo "\n* Tâche 1";};
$q[]=function() { echo "\n* Tâche 2";};
$q[]=function() { echo "\n* Tâche 3";};
$q[]=function() { echo "\n* Tâche 4";};

// Traitement de la queue
foreach ($q as $task) {
    $task();

    //Aléatoirement on ajoute la tâche à traiter à nouveau
    if (rand(0,1)==1)
        $q[] = $task;
}
?>
```

Résultats d'exécution de script

```
$ php splqueue1.php

* Tâche 1
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 1
* Tâche 4

$ php splqueue1.php

* Tâche 1
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 3

$ php splqueue1.php

* Tâche 1
```

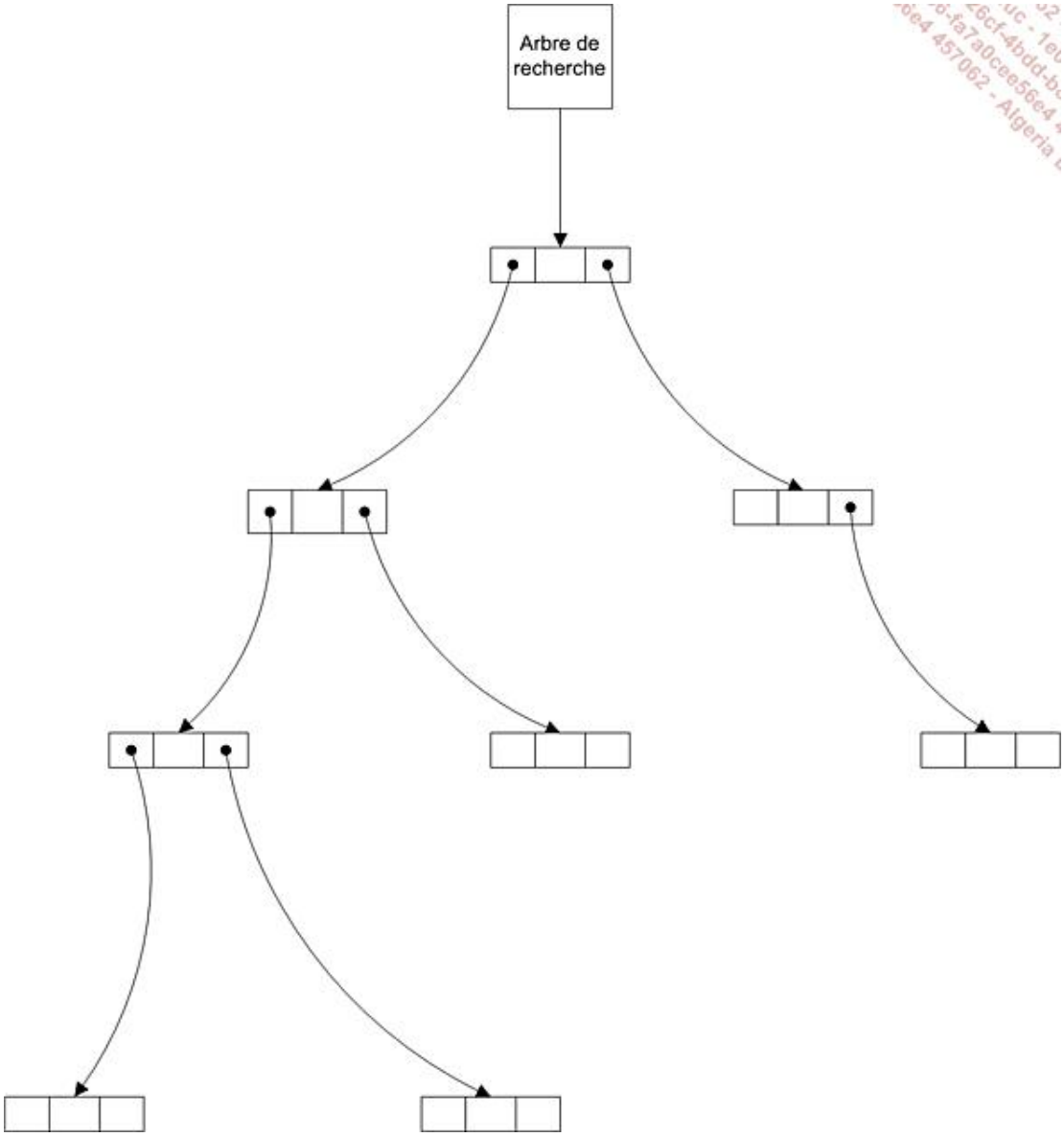

© ENI Editions - All rigths reserved - Algeria Educ

```
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 3
* Tâche 4

$ php splqueue1.php

* Tâche 1
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 1
* Tâche 1

$ php splqueue1.php

* Tâche 1
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 1
* Tâche 2
* Tâche 3
* Tâche 4
* Tâche 2
* Tâche 3
* Tâche 2
* Tâche 3
* Tâche 2
* Tâche 3
```

e. Les arbres binaires de recherche

Les arbres binaires de recherche sont des structures adaptées aux recherches optimales de données. Il faut *log(n)* opérations pour trouver et obtenir un élément dans un arbre contenant *n* éléments.

© ENI Editions - All rigths reserved - Algeria Educ

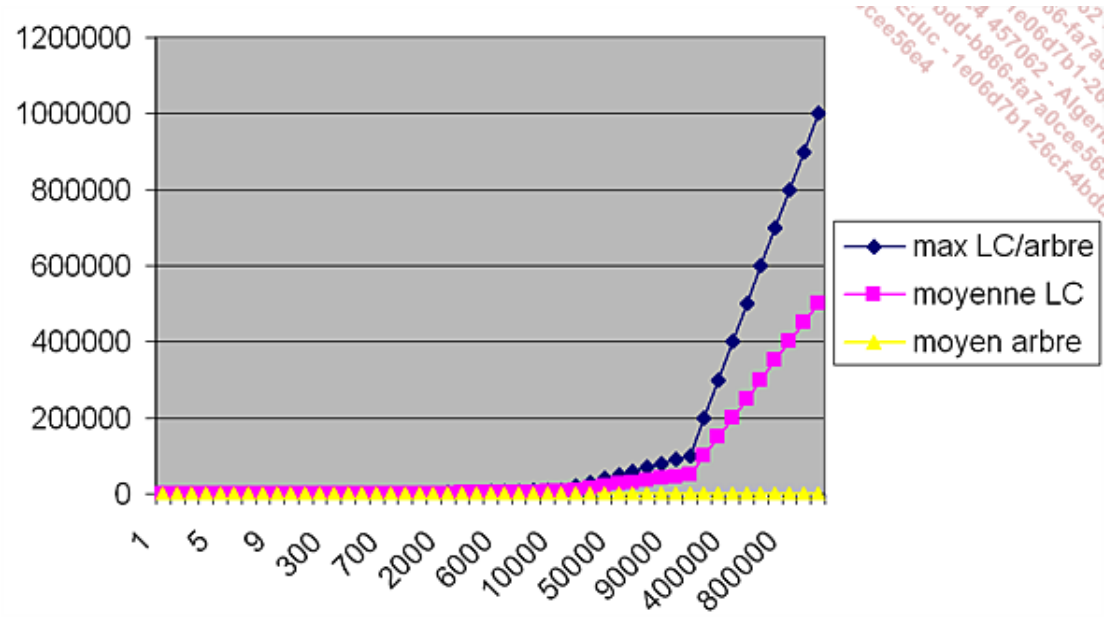


La recherche est basée sur la dichotomie qui consiste, à chaque itération de recherche, à diviser par 2 l'ensemble des résultats possibles.

Dans le cas de la liste chaînée, la recherche va prendre *n/2* opérations en moyenne et *n-1* opérations maximum.

Dans le pire des cas, l'arbre n'est pas plus rapide. Cependant, en moyenne, il est théoriquement plus rapide qu'une liste chaînée.

© ENI Editions - All rigths reserved - Algeria Educ



La recherche d'élément est plus rapide en moyenne et cela s'amplifie avec le nombre d'éléments.

f. Les arbres disponibles en PHP SPL

Il existe trois arbres distincts dans la SPL de PHP :

- SplHead
- SplMaxHeap
- SplMinHeap

Les deux dernières structures privilégient le stockage des plus grandes ou des plus petites valeurs en haut de l'arbre.

g. Exemples d'arbre SplMaxHeap

```
<?php

$abMax = new SplMaxHeap();
$abMin = new SplMinHeap();
for ($i=0;$i<20;$i++) {
    $v=rand(0, 100);
    $abMax->insert($v);
    $abMin->insert($v);
}

// Parcours arbre Max
$abMax->top();
while ($abMax->valid()) {
  echo "\n* ", $abMax->current();
  $abMax->next();
}

// Parcours arbre Min
$abMin->top();
while ($abMin->valid()) {
  echo "\n* ", $abMin->current();
  $abMin->next();
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

Résultats d'exécution de script

```
$ php splheap1.php

* 100
* 96
* 95
* 93
* 89
* 83
* 80
* 77
* 72
* 71
* 66
* 66
* 61
* 52
* 47
* 27
* 26
* 24
* 19
* 11
* 11
* 19
* 24
* 26
* 27
* 47
* 52
* 61
* 66
* 66
* 71
* 72
* 77
* 80
* 83
* 89
* 93
* 95
* 96
```

Le résultat montre clairement l'ordre mis en place dans les deux structures.

h. Les files à priorités

Une structure fort utile dans toute application devant gérer des priorités dans l'ordre de traitement des données est la classe SplPrioriyqueuee permettant d'ajouter des priorités sous forme entière au moment de l'insertion dans l'objet.

Il s'agit d'un moyen élégant et rapide de simuler un fonctionnement proche d'une administration telle que des bureaux de justice où sont privilégiées les affaires criminelles.

i. Exemple d'arbre SplPriorityQueue

Ce script permet de créer une liste aléatoire. Nous avons inséré des valeurs numériques mais rien ne nous empêche d'y insérer des morceaux de musique.

```
<?php

$pQueue = new SplPriorityQueue();

for ($i=0;$i<20;$i++) {
    $pQueue->insert($i*4, rand(0,20));
```


© ENI Editions - All rigths reserved - Algeria Educ

```
}

// Parcours arbre Max
$pQueue->top();
while ($pQueue->valid()) {
  echo "\n* ", $pQueue->current();
  $pQueue->next();
}

?>
```

Résultats d'exécution de script

```
$ php splprioqueue1.php

* 0
* 24
* 28
* 36
* 44
* 48
* 40
* 12
* 16
* 20
* 52
* 56
* 68
* 64
* 8
* 4
* 60
* 72
* 76
```

Les résultats montrent clairement la non-linéarité dans l'affichage résultant de l'ajout par priorité aléatoire.

j. Les tableaux fixes

Les tableaux fixes sont l'équivalent de tableaux PHP ayant une limite de taille.

Cela permet de limiter le nombre de places possibles comme par exemple pour l'accès à une salle de spectacle.

La classe permettant d'implémenter un tableau de taille fixe est SplFixedArray et s'utilise ensuite comme un tableau PHP classique avec l'opérateur crochet [].

```
<?php
// Tableau de taille fixe
$tabFixe = new SplFixedArray(3);
try {
    for($i=0;$i<4;$i++) {
        $tabFixe[$i] = $i*4;
    }
} catch(RuntimeException $re) {
    echo "RuntimeException: indice $i =>".$re->getMessage()."\n";
}

// On augmente dynamiquement la taille à 5
$tabFixe->setSize(5);
try {
    for($i=0;$i<4;$i++) {
        $tabFixe[$i] = $i*4;
    }
} catch(RuntimeException $re) {
    echo "RuntimeException: ".$re->getMessage()."\n";
}
//truncature des elements
$tabFixe->setSize(2);
```

© ENI Editions - All rigths reserved - Algeria Educ

```
var_dump($tabFixe);

try {
    var_dump($tabFixe['toto']);
} catch(RuntimeException $re) {
    echo "RuntimeException: indice 'toto' =>".$re->getMessage()."\n";
}

try {
    var_dump($tabFixe[-1]);
} catch(RuntimeException $re) {
    echo "RuntimeException: indice -1 =>".$re->getMessage()."\n";
}
?>
```

Résultats d'exécution de script

```
$ php splFixedArray.php
RuntimeException: indice 3 =>Index invalid or out of range
object(SplFixedArray)#1 (2) {
  [0]=>
  int(0)
  [1]=>
  int(4)
}
RuntimeException: indice 'toto' =>Index invalid or out of range
RuntimeException: indice -1 =>Index invalid or out of range
```

L'objet SplFixedArray produit une exception RuntimeException lors de dépassement de tableau ou en cas d'accès à un élément non présent.

Le traitement des accès aux éléments dans votre code doit être plus strict car un accès dans un tableau PHP via une clé inexistante ne produit qu'un message de type Notice. La classe SplFixedArray produit, quant à elle, des exceptions. Traduire brutalement vos tableaux PHP en objet SplFixedArray est donc à proscrire sans un contrôle rigoureux.

© ENI Editions - All rigths reserved - Algeria Educ

Concepts généraux de sécurité

Le concept de sécurité repose toujours sur une politique dite de sécurité.

Une politique de sécurité est un référentiel contenant l'ensemble des dispositions qu'un projet met en œuvre afin de garantir un niveau de sécurité et une garantie de service adaptée à celui-ci.

La sécurité en informatique est un concept abstrait qui peut prendre des aspects spécifiques concrets. En un mot, la sécurité n'a de sens que dans l'application de règles techniques permettant de :

- Limiter l'accès à l'information manipulée par les applications.
- Bloquer l'automatisation de récupération massive d'informations.
- Dissimuler les informations de la plate-forme technique.
- Offrir le moins de vulnérabilité aux failles de sécurité connues.
- Empêcher la récupération d'informations d'un compte utilisateur.
- Prévenir le vol d'identité simple ou en masse.
- Offrir une continuité de service élevée.
- Écarter la majorité des attaques ou tentatives de déni de service.
- Circonscrire l'utilisation des applications à un ensemble fini de fonctionnalités autorisées.
- Garantir l'exactitude de l'information.
- Garantir un temps de réponse correct pour l'ensemble des utilisateurs.

© ENI Editions - All rigths reserved - Algeria Educ

Concepts de sécurité des serveurs

La sécurisation des serveurs n'est donc pas la réponse à l'ensemble des domaines de la sécurité. Il permet cependant d'offrir des réponses ou des débuts de réponses parfois très élégantes à un certain nombre de problématiques parmi lequel :

- Dissimuler les informations de la plate-forme technique.
- Offrir le moins de vulnérabilité aux failles de sécurité connues.
- Offrir une continuité de service élevée.
- Écarter la majorité des attaques ou tentatives de déni de service.
- Circonscrire l'utilisation des applications à un ensemble fini de fonctionnalités autorisées.

© ENI Editions - All rigths reserved - Algeria Educ

Outils de base pour l'analyse des informations serveur

Il est souvent préférable d'utiliser des outils en ligne de commande car ils permettent de mieux filtrer et maîtriser l'information manipulée. Cependant, il est aussi important de posséder un ensemble d'outils faciles et rapides à utiliser dans un sens souvent naturel, du type : « J'utilise, j'analyse ». Dans cette optique, voici donc trois outils indispensables pour comprendre et percevoir les risques liés au serveur Web en lui-même.

1. Wireshark, l'analyseur de trames TCP

Un analyseur de connexions TCP permet de comprendre le concept d'attaque type man-in-the-middle. En effet, il suffit de se placer entre le serveur et le client et de récupérer toute l'information circulant « en clair » pour obtenir des informations sensibles.

L'analyseur de trames TCP, bien que rudimentaire au premier abord, donne une vision claire et définitive de ce qui circule entre le navigateur et les serveurs Web.

Il suffit d'utiliser l'analyseur Wireshark téléchargeable depuis :

http://www.wireshark.org/download.html

2. Le module Firefox Web Developer

Ce module permet de récupérer et de manipuler l'ensemble des informations d'un site :

- Les feuilles de style.
- Les formulaires.
- Les commentaires.
- Les cookies du site.
- Les images.
- Le code HTML.
- Le code JavaScript.
- Les liens.
- Les en-têtes HTTP.
- Les outils de validation.
- Divers outils pour le rendu visuel.

Il suffit d'utiliser le module Firefox Web Developer téléchargeable depuis :

https://addons.mozilla.org/fr/firefox/addon/60

Il est essentiel pour accéder à l'information de base sur les échanges d'une page.

3. Le module Firebug

Complément naturel du module précédent, le module Firebug permet de comprendre l'enchaînement des appels HTTP. Ce module considère la transaction permettant l'affichage intégral de la page comme le pivot central de l'analyse et offre tous les outils permettant d'analyser les échanges HTTP, les durées, le status et l'enchaînement de ces appels.

© ENI Editions - All rigths reserved - Algeria Educ

Sa force réside dans le fait qu'il est l'ultime moyen de détecter les erreurs d'accès et de permettre l'optimisation des pages visitées pour vos propres sites.

L'inconvénient de Firebug est qu'il ralentit fortement l'affichage des sites car il analyse le contenu en permanence.

Il suffit d'utiliser le module Firefox Firebug téléchargeable depuis :

https://addons.mozilla.org/fr/firefox/addon/1843

© ENI Editions - All rigths reserved - Algeria Educ

Limiter la transmission d'informations

Un serveur Web configuré par défaut permet d'accéder à un maximum d'informations. Pour notre exemple, un site communautaire permettant le partage de code source PHP est utilisé.

Il suffit d'utiliser l'onglet **Information > Réponse En-têtes HTTP** dans la barre Firefox Web Developer.

Voici quelques exemples de la réponse :

```
Date: Mon, 05 Apr 2010 20:34:05 GMT
Server: Apache/2.2.X (OVH)
X-Powered-By: PHP/4.4.9
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 8608
Keep-Alive: timeout=5, max=86
Connection: Keep-Alive
Content-Type: text/html

200 OK
```

Avec cette simple manipulation, nous savons que ce site utilise un serveur Apache 2.2, la plus récente version d'Apache à ce jour. Le fournisseur d'accès n'est rien d'autre qu'OVH. Le langage PHP a été utilisé pour la génération de la page. La version 4.4.9 de PHP n'a pas reçu de corrections officielles depuis le 7 août 2008. Ce qui laisse entrevoir de nombreuses possibilités pour une attaque en règle.

La sécurité par l'obscurité n'est pas une solution ultime de sécurité. Cependant, il est impératif de ne jamais donner d'information sur votre plate-forme, surtout si cette information n'offre aucune utilité.

Voici donc deux configurations à appliquer afin de limiter rapidement les informations sur les technologies et les versions utilisées par votre site Web.

1. php.ini

```
explose_php=0
```

La version de PHP et le header X-Powered-By ne seront plus visibles dans les en-têtes HTTP.

2. httpd.conf

```
ServerSignature Off
ServerTokens Prod
```

La version d'Apache ne sera plus affichée dans les en-têtes ni dans les pages d'erreur.

3. Les balises META

Les balises META de votre code HTML donnent parfois des informations sur un éventuel CMS (application PHP de gestion de contenu) utilisé pour créer et gérer votre site Web. L'ensemble des CMS possède de nombreuses failles de sécurité.

Un exemple avec CMS Made Simple :

```
<meta name="Generator" content="CMS Made Simple - Copyright (C)
2004-9 Ted Kulp. All rights reserved." />
```

Un second exemple pour un site créé avec Joomla version 1.5

```
<meta name="generator" content="Joomla! 1.5 - Open Source Content
```

© ENI Editions - All rigths reserved - Algeria Educ

```
Management" />
```

Il est tout simplement préconisé de les retirer. Vos lecteurs n’auront aucun avantage à connaître les technologies utilisées pour votre site Web. Cependant, une personne mal intentionnée a un intérêt particulier pour ce type d’information qui lui permet de cibler les vulnérabilités associées aux technologies, logiciels et versions utilisés pour votre service ou site Internet.

© ENI Editions - All rigths reserved - Algeria Educ

Réécriture d'URL pour masquer PHP

Le principe de la réécriture d'URL est de permettre de transformer une URL en une autre au niveau du serveur.

Cela permet de ne jamais dévoiler ni la technologie utilisée ni le nom des scripts PHP utilisés.

Il est aussi parfois intéressant d'utiliser un site et de le copier complètement afin d'en avoir une copie sur votre ordinateur toujours à disposition. Cette technique s'appelle l'aspiration de site.

Un bon logiciel libre permettant de réaliser cette tâche est *HTTract website copier* disponible à l'URL suivante : http://www.httrack.com/page/2/fr/index.html

L'exemple le plus courant d'utilisation de la réécriture d'URL est de créer un site et de l'aspirer ensuite. Votre script PHP génère des pages HTML à partir d'un paramètre et le serveur Web renvoie des pages HTML avec le nom du paramètre.

http://monsite.eni.fr/index.html peut être traduit par http://monsite.eni.fr/script.php?param=index

À partir de cette étape, il est possible d'aspirer le site afin de le mettre sur clé USB, sur un serveur Web sans PHP et de déposer des pages statiques simples.

Le seul point important est que le script PHP produise du code HTML avec les liens vers des URL avant une extension .html. Cela signifie qu'il y a une cohérence entre le script et les règles de réécriture.

Dans la plupart des CMS utilisant la réécriture, le choix de la forme des liens est disponible au travers d'un paramètre de configuration, de sorte qu'un hébergeur ne vous offrant pas cette fonctionnalité ne soit pas inutilisable.

1. Activation du module Apache de réécriture

Le module et les directives de configuration seront actifs dès que la directive de chargement du module sera déclarée.

```
LoadModule rewrite_module modules/mod_rewrite.so
```

2. Exemple de réécriture d'appel html en appel php

```
RewriteEngine on
RewriteRule ([^.]+)\.html$ /index.php?page=$1 [L]
```

Voila comment, en trois lignes de configuration, tous vos appels sur votre site n'auront plus jamais l'aspect d'URL de téléchargement et comment vous pourrez, avec un peu de patience, créer des URL reflétant de manière simple et élégante le contenu de votre page.

3. Exemple de réécriture avec système basique de cache

Il est possible d'écrire un script générant des pages temporaires.

```
RewriteEngine on

RewriteCond %{REQUEST_FILENAME}\.tmp -f
RewriteRule ([^.]+)\.html$ /$1.html.tmp [L]

RewriteRule ([^.]+)\.html$ /page.php?page=$1 [L]
```

La configuration indique que, s'il existe un fichier .html.tmp, alors on réécrit la requête vers ce fichier, sinon on réécrit l'URL vers le script PHP.

4. En cas de problème

Il est possible d'activer les traces de l'extension de réécriture. Pour cela, vous devez déclarer dans la configuration du serveur ou, mieux encore, dans la déclaration de votre hôte virtuel, les informations suivantes :

```
RewriteLog logs/rewrite.log
```

© ENI Editions - All rigths reserved - Algeria Educ

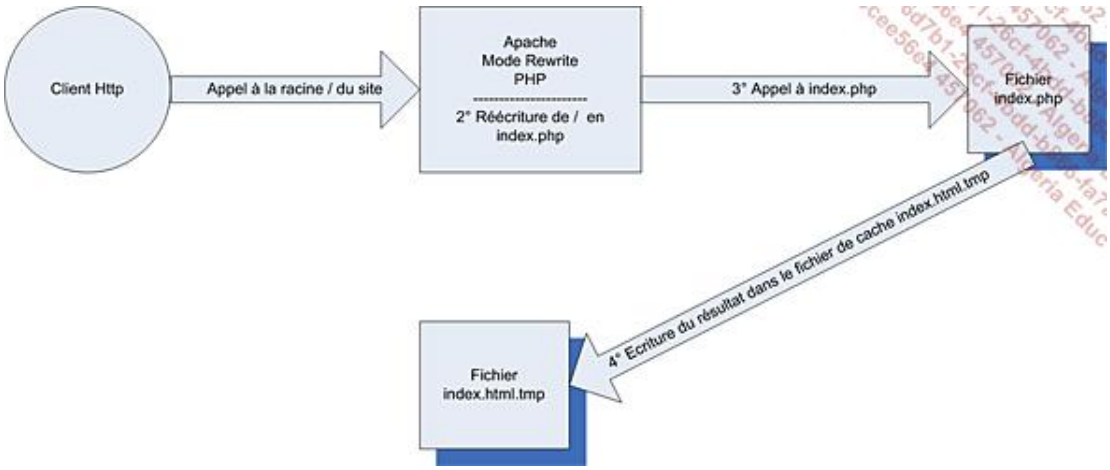
```
RewriteLogLevel 3
```

Après un rechargement ou un redémarrage, Apache remplira un fichier rewrite.log dans son répertoire de log et ce fichier contiendra l'intégralité des informations de réécriture.

5. Exemple complet

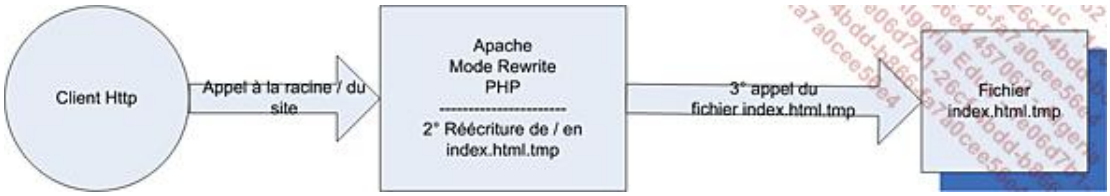
a. Principe de base

L'idée de cet exemple est de permettre de consulter des pages HTML en consultant des URL avec une extension .html, tout en ayant la capacité de générer ces pages à la volée.



Le premier appel au site doit produire les étapes suivantes :

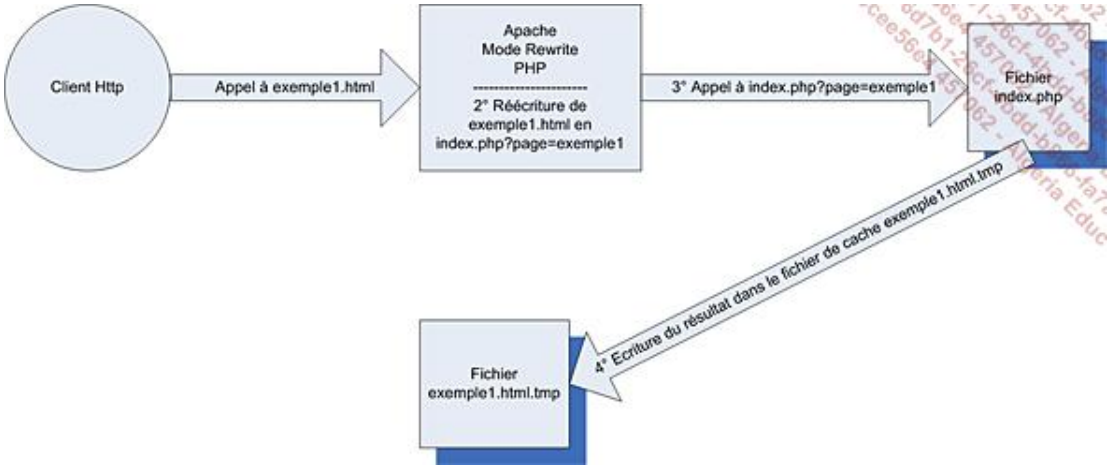
- Appel de : http://rewrite.eni.fr/.
- Recherche de la page : index.html.tmp.
- Échec de recherche du fichier index.html.tmp.
- Recherche de la page : index.php.
- Exécution du script index.php.
- Génération de la page index.html.tmp par le script PHP index.php.
- Renvoi du contenu.



Le second appel au site doit produire les étapes suivantes :

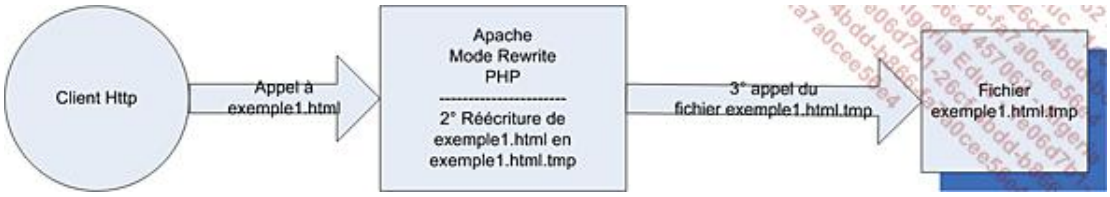
- Appel de : http://rewrite.eni.fr/.
- Recherche du fichier : index.html.tmp.
- Renvoi du contenu du fichier index.html.tmp.

© ENI Editions - All rigths reserved - Algeria Educ



Le premier appel à une page avec extension .html :

- Appel de : http://rewrite.eni.fr/exemple1.html
- Recherche de la page : exemple1.html.tmp
- Échec de recherche du fichier exemple1.html.tmp
- Analyse et création de la page d'appel
- Recherche de la page : index.php
- Exécution du script : index.php?page=exemple1
    - Le paramètre page est le nom appelé sans l'extension .html
    - Page vaut exemple1 ici
- Génération de la page exemple1.html.tmp par le script PHP index.php
- Renvoi du contenu



Le second appel à une page avec extension .html :

- Appel de : http://rewrite.eni.fr/exemple1.html
- Recherche du fichier : exemple1.html.tmp
- Renvoi du contenu du fichier exemple1.html.tmp

Cette technique de réécriture est particulièrement efficace car, en cas de second appel, l'interpréteur PHP n'est pas du tout utilisé et permet donc d'atteindre les performances de consultation de pages statiques, alors même que notre site a été généré dynamiquement. Il ne s'agit pas là d'une solution miracle. Cependant, si une grande partie de vos pages ont un contenu qui ne varie que très peu, alors ce type d'approche peut vous permettre de gagner en performance, surtout pour des sites très consultés, et donc de réduire massivement le nombre de serveurs dédiés à la consultation des pages.

© ENI Editions - All rigths reserved - Algeria Educ

b. Configuration générale

Cette directive permet simplement de compléter la configuration par répertoire. Ainsi chaque répertoire peut disposer d'une configuration Apache spécifique. Ici, on autorise simplement la configuration de l'ensemble des éléments paramétrables par Apache.

```
AllowOverride All
```

La directive générale permet de choisir une configuration spécifique par répertoire. Souvent, le fichier de directive se nomme .htaccess mais peut très bien être renommé par la directive générale AccessFileName.

c. Configuration spécifique

Pour notre exemple, notre fichier de directive cherche une version de cache appropriée à l'appel et, si celle-ci n'existe pas, découpe l'URL demandée et appelle le script avec le bon paramètre page.

```
# On prend le cache index.html.tmp par défaut sinon le script
DirectoryIndex index.html.tmp index.php
# activation du moteur de réécriture
RewriteEngine on
# Si le fichier de cache existe, il est utilisé
RewriteCond %{REQUEST_FILENAME}\.tmp -f
RewriteRule ([^.]+)\.html$ $1.html.tmp [L]

#sinon on appelle le script avec le paramètre adéquat
RewriteRule ([^.]+)\.html$ index.php?page=$1 [L]
```

d. Le script de génération des pages

Ce script permet de :

- générer le contenu d'une page la première fois,
- créer le fichier de cache pour chaque page,
- supprimer les caches par recherche.

```
<?php
echo "<h3>Dans le script PHP</h3>";

// si aucun paramètre n'est fourni on prend index par défaut
if (isset($_GET['page'])) $param=$_GET['page'];
else $param="index";

// si le paramètre passé est 'reset', on supprime l'ensemble des
caches
if ($param=="reset") {
    $nb=viderLesCaches();
    echo "</h1>Vidage du cache réalisé - $nb cache(s)
supprimé(s).</h1>";
    exit(0);
}

// création du nom du fichier de cache
$file_page=$param.".html.tmp";
$data="<h1>$param</h1>\nCeci est la page de $param.";

// suppression du cache
if (file_exists($file_page)) unlink($file_page);

// Ecriture du fichier de cache avec pied de page avec la date de
génération
```


© ENI Editions - All rigths reserved - Algeria Educ

```
file_put_contents($file_page, "$data<br\><pre>mis en cache le
".date("F j, Y, g:i a")."</pre>");

//envoi des données
echo $data;


// Une fonction pour supprimer les caches existants
function viderLesCaches() {
    $i=0;

    if ($dh = opendir(".")) {
        while (($file = readdir($dh)) !== false) {
            if (is_file($file) and substr($file, -3,
3)=="tmp") {
                unlink($file);
                $i++;
            }
        }
        closedir($dh);
    }
    return $i;
}
?>
```

e. Activation des traces de réécriture

L'activation des traces de réécriture permet :

- Le suivi des appels.
- Le suivi de l'enchaînement des étapes de la réécriture.
- Une meilleure compréhension du fonctionnement.

```
RewriteLogLevel 3
RewriteLog "logs/rewrite.log"
```

Cependant, l'activation des traces sur l'étape de réécriture pose un problème de lenteur dû à trois éléments majeurs :

- La collecte des informations.
- L'écriture dans un fichier d'information.
- Le volume lié au nombre d'appels.

Il est donc déconseillé, hors des environnements de développement, d'activer ce type de directive sans mesurer l'impact sur le rendu de service et la performance.

f. Traces liées à l'appel sans cache

```
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'mx.html'
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (3) [perdir
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/mx.html -> mx.html
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (3) [perdir
```

© ENI Editions - All rigths reserved - Algeria Educ

```
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'mx.html'
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (2) [perdir
C:/wamp/www/rewrite/] rewrite 'mx.html' -> 'index.php?page=mx'
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (3) split
uri=index.php?page=mx -> uri=index.php, args=page=mx
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (3) [perdir
C:/wamp/www/rewrite/] add per-dir prefix: index.php ->
C:/wamp/www/rewrite/index.php
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (2) [perdir
C:/wamp/www/rewrite/] strip document_root prefix:
C:/wamp/www/rewrite/index.php ->/rewrite/index.php
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1db5860/initial] (1) [perdir
C:/wamp/www/rewrite/] internal redirect with /rewrite/index.php
[INTERNAL REDIRECT]
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1deeeb0/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/index.php -> index.php
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1deeeb0/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'index.php'
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1deeeb0/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/index.php -> index.php
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1deeeb0/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'index.php'
127.0.0.1 - - [11/Apr/2010:08:41:10 +0200]
[localhost/sid#8e5150][rid#1deeeb0/initial/redir#1] (1) [perdir
C:/wamp/www/rewrite/] pass through C:/wamp/www/rewrite/index.php
```

g. Traces liées à l'appel avec cache

```
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (3) [perdir
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/index.html -> index.html
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'index.html'
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (2) [perdir
C:/wamp/www/rewrite/] rewrite 'index.html' -> 'index.html.tmp'
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (3) [perdir
C:/wamp/www/rewrite/] add per-dir prefix: index.html.tmp ->
C:/wamp/www/rewrite/index.html.tmp
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (2) [perdir
C:/wamp/www/rewrite/] strip document_root prefix:
C:/wamp/www/rewrite/index.html.tmp -> /rewrite/index.html.tmp
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1dc3898/initial] (1) [perdir
C:/wamp/www/rewrite/] internal redirect with /rewrite/index.html.tmp
[INTERNAL REDIRECT]
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1ec1400/initial/redir#1] (3) [perdir
```


© ENI Editions - All rigths reserved - Algeria Educ

```
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/index.html.tmp -> index.html.tmp
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1ec1400/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'index.html.tmp'
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1ec1400/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] strip per-dir prefix:
C:/wamp/www/rewrite/index.html.tmp -> index.html.tmp
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1ec1400/initial/redir#1] (3) [perdir
C:/wamp/www/rewrite/] applying pattern '([^.]+)\.html$' to uri
'index.html.tmp'
127.0.0.1 - - [11/Apr/2010:08:39:40 +0200]
[localhost/sid#8e5150][rid#1ec1400/initial/redir#1] (1) [perdir
C:/wamp/www/rewrite/] pass through
C:/wamp/www/rewrite/index.html.tmp
```

h. Réduction du volume des traces

Nous avons vu que le volume peut vite devenir trop important, même au niveau 3.

Une réduction au niveau 1 masque de nombreuses étapes. En effet, le niveau 1 ne permet plus de voir l'ordre, les règles et le résultat de chaque étape de la réécriture.

```
RewriteLogLevel 1
RewriteLog "logs/rewrite.log"
```

```
127.0.0.1 - - [11/Apr/2010:08:49:31 +0200]
[localhost/sid#895150][rid#1c731e8/initial] (1) [perdir
C:/wamp/www/rewrite/] internal redirect with /rewrite/mx.html.tmp
[INTERNAL REDIRECT]
127.0.0.1 - - [11/Apr/2010:08:49:31 +0200]
[localhost/sid#895150][rid#1cb9390/initial/redir#1] (1) [perdir
C:/wamp/www/rewrite/] pass through C:/wamp/www/rewrite/mx.html.tmp
127.0.0.1 - - [11/Apr/2010:08:49:48 +0200]
[localhost/sid#895150][rid#1c7d210/initial] (1) [perdir
C:/wamp/www/rewrite/] internal redirect with /rewrite/index.php
[INTERNAL REDIRECT]
127.0.0.1 - - [11/Apr/2010:08:49:48 +0200]
[localhost/sid#895150][rid#1d87320/initial/redir#1] (1) [perdir
C:/wamp/www/rewrite/] pass through C:/wamp/www/rewrite/index.php
127.0.0.1 - - [11/Apr/2010:08:49:50 +0200]
[localhost/sid#895150][rid#1c81220/initial] (1) [perdir
C:/wamp/www/rewrite/] internal redirect with
/rewrite/mdddx.html.tmp [INTERNAL REDIRECT]
127.0.0.1 - - [11/Apr/2010:08:49:50 +0200]
[localhost/sid#895150][rid#1d8aa10/initial/redir#1] (1) [perdir
C:/wamp/www/rewrite/] pass through C:/wamp/www/rewrite/mdddx.html.tmp
```

© ENI Editions - All rigths reserved - Algeria Educ

Limiter la transmission des erreurs

Il existe deux moyens de limiter la transmission des erreurs.

Premièrement, il faut interdire l'affichage des erreurs dans les pages de résultats d'exécution de vos scripts PHP. En effet, les messages d'erreur révèlent trop d'informations sur vos versions applicatives, telles que :

- le nom des logiciels,
- la version des logiciels,
- la version du moteur PHP.

Les technologies de stockage peuvent être mises en évidence telles que :

- Nom de fichier accessible directement depuis votre serveur.
- MySQL et sa version.
- Oracle et sa version.
- SQLite et sa version.

Deuxièmement, il faut définir les pages par défaut en cas d'erreur. Avec un peu de malice et quelques appels à des pages inexistantes, vous obtenez rapidement les informations sur les versions d'Apache, sur les modules activés et leur version et aussi sur les technologies sous-jacentes aux services offerts. Une deuxième stratégie consiste à appeler un script PHP avec des paramètres erronés afin de créer une erreur interne. Cette dernière technique est plus difficile à réaliser mais peut, cependant, apporter de nombreuses informations utilisables.

Limiter l'affichage des erreurs PHP

Ces paramètres sont à placer dans le fichier php.ini de configuration de votre serveur.

```
display_errors = Off
display_startup_errors = Off
```

Ici, on interdit formellement l'affichage des erreurs d'analyse et de lancement du script PHP ainsi que toute erreur survenant lors de l'exécution du code (l'affichage des paramètres d'accès à une base de données, par exemple).

Rediriger les pages d'erreur Apache

Ces directives sont à placer dans votre configuration de serveur Apache.

```
ErrorDocument 500 "Erreur Interne"
ErrorDocument 404 «/page_non_trouve.html»
ErrorDocument 401 « Erreur authentification »
ErrorDocument 403 "Pas de droits d'accès à cette page"
```

Ici, le but est de ne rien laisser filtrer sur les pages d'erreur de vos applications afin que ne puissent pas être dévoilés les versions applicatives, le choix des technologies et les informations sur l'accès aux données.

© ENI Editions - All rigths reserved - Algeria Educ

Limiter la part de trafic alloué à chaque utilisateur

Le module mod_bw permet de limiter la bande passante de votre serveur, de garantir, pour chaque utilisateur, un service de qualité et d'éviter d'offrir le moyen à une minorité de vos utilisateurs d'écrouler vos ressources techniques par des téléchargements trop importants.

Disponible sous Windows comme dans la plupart des distributions Linux, mod_bw s'utilise comme un module classique et est téléchargeable librement depuis l'adresse : http://bwmod.sourceforge.net/.

Les trois paramètres principaux :

L'option Bandwidth permet de définir une bande passante maximum par connexion.

L'option Minbandwidth permet de garantir une bande passante minimum par connexion.

L'option LargeFileLimit permet de définir une limite de bande passante par type de fichier.

Voici quelques exemples d'utilisation du module Apache.

Limiter la bande passante à 200 Ko/s

```
<Virtualhost *>
<IfModule mod_bw.c>
BandwidthModule On
ForceBandWidthModule On
Bandwidth all 200240
MinBandwidth all -1
</IfModule>
Servername php.eni.fr
</Virtualhost>
```

Allouer une bande passante minimum à 50 Ko/s

```
<Virtualhost *>
<IfModule mod_bw.c>
BandwidthModule On
ForceBandWidthModule On
MinBandwidth all 50000
</IfModule>
Servername php.eni.fr
</Virtualhost>
```

Limiter la bande passante des fichiers zip à 20 Kb/s

```
<Virtualhost *>
<IfModule mod_bw.c>
BandwidthModule On
ForceBandWidthModule On
LargeFileLimit .zip 1 20000
</IfModule>
Servername php.eni.fr
</Virtualhost>
```

© ENI Editions - All rigths reserved - Algeria Educ

Limiter l'accès à certaines URL de l'application

Il faut distinguer deux types de limite d'accès, les accès limités préventivement et les accès limités par rejet. Grâce aux accès limités préventivement, vous signalez que vous ne souhaitez pas que certaines parties de votre site soient visitées, entre autres par les robots d'indexation des moteurs de recherche. Dans les accès limités par rejet, vous paramétrez votre serveur afin d'interdire l'accès aux répertoires et aux fichiers.

1. Le fichier robots.txt

Le fichier robots.txt est un fichier abordé par tous les robots d'indexation modernes. Il indique ce que vous souhaitez voir indexer dans les pages de réponse des moteurs de recherche et ce que vous ne souhaitez pas voir apparaître.

Restriction totale par robots.txt

```
User-agent: *
Disallow: /
```

Ce fichier indique qu'aucun contenu ne doit être indexé par aucun robot d'indexation.

Restriction partielle par robots.txt

```
User-agent: *
Disallow: /admim/*
Disallow: /images/*
Disallow: /videos/*
```

Ce fichier indique que le contenu est indexable intégralement à l'exception du contenu de trois répertoires : /admim, /images, /videos.

Autoriser uniquement Google à indexer par robots.txt

```
User-agent: Google
Disallow:

User-agent: *
Disallow: /
```

2. Le module de contrôle d'accès

Le contrôle d'accès est très complet. Il permet de restreindre l'accès selon plusieurs critères :

- Restriction sur répertoire.
- Restriction par IP ou nom de domaine appelant.
- Restriction par mot de passe.
- Restriction par méthode HTTP.

L'authentification afin de restreindre l'accès peut s'appuyer sur de nombreuses sources de données :

- Fichier .htpasswd.
- Base LDAP.
- Base MySQL.

© ENI Editions - All rigths reserved - Algeria Educ

Exemple de restriction totale

```
<Directory /var/www/html/eni>
Order allow,deny
Deny from all
</Directory>
```

Exemple de restriction par IP

```
<Directory /var/www/html/eni_interne>
Order deny,allow
Deny from all
Allow 192.168.10.1
</Directory>
```

Ici, seul le client ayant une IP 192.168.10.1 pourra accéder aux répertoires /eni_interne

Exemple de restriction par mot de passe

```
<Directory /var/www/html/eni_utilisateur>
AuthUserFile .htpasswd
AuthGroupFile /dev/null
AuthName "Veuillez vous identifier"
AuthType Basic

<Limit GET POST>
require valid-user
</Limit>
</Directory>
```

Il est important de positionner le fichier des groupes à /dev/null, sinon le fichier des utilisateurs .htpasswd ne sera pas pris en compte et aucune authentification ne sera validée.

Génération du fichier de mot de passe

```
# htpasswd -b .htpasswd jmren totozero00
```

Ici le fichier .htpasswd correspond au fichier de la directive AuthUserFile et doit donc être présent dans le répertoire /var/www/html/eni_utilisateur.

Exemple de restriction par LDAP

```
LoadModule dav_svn_module     modules/mod_dav_svn.so
LoadModule authz_svn_module   modules/mod_authz_svn.so

LDAPTrustedGlobalCert CA_BASE64 /etc/openldap/cacerts/ca.pem
LDAPVerifyServerCert On
LDAPCacheEntries 1024
LDAPCacheTTL 600
LDAPOpCacheEntries 1024
LDAPOPCacheTTL 600
LDAPSharedCacheFile /tmp/ldap_apache_cache

<Directory "/var/www/html/eni_ldap/" >
   AuthName "Authentification LDAP"
   AuthType Basic
   AuthBasicProvider ldap
   AuthLDAPURL ldaps://ldap-
server/dc=auth,dc=eni,dc=fr?uid?sub?ou=Utilisateur
   AuthzLDAPAuthoritative off
   require valid-user
```


© ENI Editions - All rigths reserved - Algeria Educ

```
</Directory>
```

Ici, le plus complexe est de comprendre que, pour une sécurité optimale, il est important de garantir une sécurité des échanges réseau cryptés de bout en bout.

Cela explique pourquoi l'exemple de configuration ne concerne que la mise en place d'une configuration LDAP sécurisée ou LDAPS.

Compromettre la sécurité d'un mot de passe via LDAP consiste à potentiellement remettre en cause la sécurité de l'ensemble du système d'information car le serveur LDAP va permettre de centraliser l'identité des utilisateurs et donc de compromettre, non pas l'accès à un service, mais à un ensemble de service.

Il faut bien mesurer le risque de la centralisation de l'authentification et replacer cette démarche dans un ensemble cohérent lié à une politique de sécurité qu'il faudra respecter et appliquer.

© ENI Editions - All rigths reserved - Algeria Educ

Réduire l’utilisation aux trois modes de base

Il existe trois modes classiques dans le protocole http : POST, GET et HEAD.

Il existe cependant trois autres modes moins connus : OPTIONS, TRACE, CONNECT. Des attaques peuvent être menées au travers des trois derniers modes cités. Pour éviter cela, il n’y a rien de mieux que de les interdire. Pour cela, deux configurations sont possibles. La première ne limitant que le mode TRACE alors que la seconde limite l’accès aux trois modes : TRACE, CONNECT et OPTIONS d’Apache.

Désactivation du mode TRACE

```
TraceEnable off
```

Désactivation des modes sensibles par configuration

```
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^(TRACE|OPTIONS|CONNECT)
RewriteRule .* - [F]
```

© ENI Editions - All rigths reserved - Algeria Educ

Restreindre les droits de configuration

La plupart des hébergeurs donnent la possibilité de paramétrer chaque répertoire au travers d'un fichier spécifique nommé par défaut .htaccess.

Afin de ne pas généraliser la lecture systématique du fichier .htaccess dans chaque répertoire, il est possible de restreindre fortement les possibilités de paramétrage individuel de chaque répertoire afin d'optimiser les accès aux fichiers en consultation et d'augmenter la sécurité en ne permettant plus les configurations utilisateur. Afin de restreindre cette possibilité, il suffit de positionner les paramètres Options et AllowOverride à None.

Exemple de configuration restrictive

```
<VirtualHost *:80>
DocumentRoot /var/www/html/monsite
<Directory "/var/www/html/monsite">
#Options Indexes FollowSymLinks
Options None
AllowOverride None
Order allow,deny
Allow from all
</Directory>
</VirtualHost>
```

© ENI Editions - All rigths reserved - Algeria Educ

Désactivation de la validation de fichier

Les Etag sont des en-têtes HTTP contenant une chaîne arbitrairement générée par le serveur afin d'identifier le fichier et détecter ses modifications éventuelles.

Désactiver la génération des en-têtes Etag

```
Header unset ETag
FileETag None
```

1. Quels sont les impacts de la désactivation des en-têtes Etag ?

Le fait de retirer les en-têtes HTTP Etag enlève la possibilité aux caches Web et aux navigateurs de valider les fichiers. Ces caches et navigateurs devront donc s'appuyer uniquement sur les deux en-têtes HTTP If-Modified-Since et If-None-Match indiquant une durée de vie maximum pour la ressource initialement reçue.

L'en-tête HTTP Etag est le seul mécanisme permettant de savoir si une nouvelle version de la ressource (fichiers) est disponible sur serveur.

2. Quels sont les risques de laisser l'option activée ?

La valeur de l'en-tête Etag est calculée sur la base du contenu, la somme de contrôle Md5, la taille et la date de dernière modification. Il est donc possible de saturer le serveur en lui envoyant de nombreux appels sur la même ressource en forçant le serveur à recalculer la valeur de l'en-tête.

Chaque appel entraînant un calcul de chaîne et des appels au système de fichier, il est possible d'obtenir un déni de service en envoyant de très nombreuses requêtes sur une même ressource. Le serveur se met à calculer et à accéder aux propriétés d'un fichier en particulier et provoque une saturation sur l'accès à une seule et même ressource, ce qui va provoquer des mises en attente de nombreux autres appels et engendrera une dégradation forte du service.

Il est donc préférable de désactiver cette option d'une utilité très relative et, peut-être, choisir une durée de vie de cache plus courte grâce à l'en-tête Last-Modified.

© ENI Editions - All rigths reserved - Algeria Educ

Changer le propriétaire des fichiers

Le changement de propriétaire des fichiers est très important. Il ne faut absolument pas positionner le propriétaire du fichier sur le compte de lancement du serveur Web. Par exemple, un serveur Apache sous Linux lancé avec le compte HTTP et le groupe HTTP ne doit pas accéder à des répertoires ni à des fichiers ayant HTTP comme propriétaire ni ayant HTTP comme groupe.

Il est parfois envisageable de créer un répertoire et d'y offrir tous les droits en lecture et en écriture pour tous les utilisateurs. Si vos scripts PHP génèrent des fichiers dans ces répertoires, ils auront comme propriétaire HTTP et comme groupe HTTP. Cependant, un attaquant ayant usurpé tous les droits sur le serveur web n'aura pas d'autres droits que celui de supprimer, au pire, l'ensemble des fichiers et des répertoires générés par vos applications. Les fichiers et répertoires de votre application n'appartenant pas au même utilisateur ni au même groupe, ne pourront être supprimés ou modifiés à votre insu et servir des intérêts peu louables.

Ne pas déployer les fichiers de vos applications avec le même propriétaire ou le même groupe que le compte de lancement de votre serveur Web.

© ENI Editions - All rigths reserved - Algeria Educ

Suppression des répertoires inutiles

Inutile de vous parler de sites ayant une page par défaut donnant accès à la documentation du serveur Web, sa version, ou le système d'exploitation hébergeant le serveur Web. Le manuel du serveur est souvent le prétexte, pour un attaquant, de choisir une cible facile. En effet, la présence par défaut de la documentation ou de la page de bienvenue est souvent le signe révélateur d'un désintérêt ou d'une méconnaissance du paramétrage du serveur Web. Cela reste aussi la meilleure des invitations possibles. En effet, un attaquant va-t-il s'attaquer à un site installé basiquement ou à un site ayant suivi l'ensemble des préconisations de sécurisation présentées par l'ensemble des acteurs sérieux du monde de la sécurité ?

User de psychologie hier comme aujourd'hui reste encore un bon moyen de défense face aux attaques.

Pour cela, supprimer les répertoires d'installation par défaut (/manual/, /icons/, etc.) et retirer toutes les configurations de page d'accueil par défaut.

© ENI Editions - All rigths reserved - Algeria Educ

Désactivation des modules inutiles dans Apache

Avec le temps, les serveurs Web se sont enrichis de modules permettant d'améliorer leur comportement et d'unifier plusieurs fonctionnalités basiques mais importantes autour des échanges Web (fonctionnalités telles que le FTP, le relais HTTP, l'accès aux fichiers, le système de versioning, etc.).

L'ensemble des modules non utilisés doit être désactivé en commentant les lignes de chargement de module. Elles commencent toutes par la directive LoadModule.

Il va en résulter deux avantages majeurs :

- L'incapacité d'exploiter des failles de module.
- La réduction de l'empreinte mémoire de chaque processus.

Le module mod_proxy, par exemple, permet de transformer votre serveur Web en relais de requête HTTP vers le Web s'il est mal paramétré. En effet, ce module a une réelle fonctionnalité pour rediriger vos requêtes vers des serveurs d'applications dédiés, cependant il ne doit pas servir de relais ouvert vers le Web.

Chaque module étant chargé dans le processus, il est d'autant plus long à charger et utilise plus de ressources physiques (RAM). Il est donc important de travailler cela afin d'obtenir des performances qui, ajoutées à l'ensemble des efforts d'optimisation, permettent d'offrir une vraie qualité de service pour vos sites Web, applications PHP ou services en ligne.

Désactiver le module proxy et proxy_ftp

```
#LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
#LoadModule proxy_module modules/mod_proxy.so
```

© ENI Editions - All rigths reserved - Algeria Educ

Différents types d’attaque possibles

Il existe deux types d’abus : l’abus de confiance du client et l’abus de confiance du serveur.

Dans le premier cas, le client et son navigateur utilisent un service en ligne, croyant être sur un site authentique ou légitime, mais se retrouvent sur une copie d’un site qui souhaite extraire et collecter des informations telles que : numéros des cartes de crédit, e-mail, informations personnelles, etc.

Le second type d’abus, plus fréquent, consiste à abuser de la confiance que les sites Web accordent à leurs utilisateurs. Plus fréquent qu’on ne le pense, ce type d’abus s’appuie sur l’ouverture de droits et d’accès plus importants.

Il est plus rare de trouver des abus visant à nuire à la fois à l’utilisateur et au site Web. Il se peut cependant que ce type d’acte nuise au site par une atteinte à l’image d’une entreprise ou d’une marque.

Il existe deux types d’attaques : les attaques sociales ou technologiques. Elles peuvent provenir de deux sources différentes : interne au personnel gestionnaire du site ou externe (personne tierce à l’entreprise).

Les abus sociaux sont plus difficiles à prévenir car chacun est amené à aider ses partenaires à faire évoluer ses solutions techniques et, dans un contexte de vagues technologiques, une confiance entre les personnes est indispensable à la réussite des projets sur Internet.

Les abus internes sociaux sont donc les plus difficiles à parer, et sans un budget conséquent, il n’existe que peu de moyens efficaces permettant de limiter l’accès aux serveurs physiques tout en contrôlant finement qui se connecte, qui maintient le code source à jour, qui le relie, etc.

Ce chapitre n’a pas la volonté de vouloir vous transformer en expert en délation et en espionnage professionnel. Il a simplement pour but de vous sensibiliser aux moyens mis à votre disposition afin d’éviter les abus technologiques externes visant spécifiquement votre ou vos serveurs Web.

Nous verrons ensemble comment mettre en place des moyens efficaces garantissant à coup sûr une sécurité optimale (utilisation de bonnes pratiques de sécurité).

Il existe de nombreuses sources d’information de qualité sur Internet, je tiens à citer en exemple deux d’entre elles, de par leur complétude et leur simplicité.

Le guide de la sécurité PHP : http://phpsec.org/projects/guide/

Les 25 erreurs de sécurité de programmation : http://www.sans.org/top25-programming-errors/

© ENI Editions - All rigths reserved - Algeria Educ

Sécurisation des accès aux pages classiques

Les paramètres en trop doivent être ignorés et invalidés

http://www.eni.fr/carnet?authentification=vrai

http://www.eni.fr/carnet?utilisateur=eni&superadmin=vrai&authentification=vrai

En effet, il est important de fournir les meilleures réponses à tout moment et de ne jamais être un intégriste des paramètres « en trop ».

Si un utilisateur utilise un paramètre en trop, ignorez l'utilisation du paramètre supplémentaire. Au bout de plusieurs tentatives, l'attaquant comprendra que vous anticipez ce type d'utilisation.

Toutes vos pages doivent valider les paramètres utilisés en entrée

Pour les paramètres restants, valides, il est important de vérifier le format du paramètre. Par exemple, si vous attendez un nombre de réponses possibles, il est important que vous validiez la présence d'un nombre et pas d'une chaîne de caractères. L'idée d'une attaque est simple : utilisation non conventionnelle, génération d'erreurs, récupération d'informations voire affaiblissement du système.

La validation du format des paramètres est importante

Le contenu peut parfois être correct mais contenir des valeurs suspectes qu'il convient de limiter, de reformater ou tout simplement de bannir.

En effet, le format le plus difficile à définir comme suspect est la chaîne de caractères.

Un attaquant peut tenter de passer du code malicieux pour des bases de données, des fichiers, du JavaScript, etc.

Il est donc important de bien définir le risque et de se prémunir des aspects les plus critiques d'un contournement de fonctionnalité par trucage de paramètre.

L'idée la plus simple est de définir des domaines de valeurs très stricts grâce à des expressions régulières limitant fortement l'utilisation des caractères nécessaires au langage de programmation et de traduire tous les caractères spéciaux en les échappant ou en y ajoutant un \ devant afin de neutraliser leur interprétation.

Un autre aspect de l'attaque de formulaire consiste tout simplement à utiliser les paramètres et leur valeur standard et à les associer différemment afin d'obtenir des informations ne vous concernant pas.

Exemple d'appel à la limite de validité

http://ecole.eni.fr/informationCompte.php?authentification=vrai&**utilisateur=unautreutilisateur**

Ici, rien ne nous interdit l'utilisation des paramètres tels quels. Cependant, il y a une tentative de contournement évidente de l'utilisation du script ou du service mis à disposition pour usurper des informations sur un tiers utilisateur.

Définir des domaines de valeurs stricts pour vos variables et protéger les caractères spéciaux des commandes sont les seuls moyens génériques de garantir une sécurité optimale à votre code.

Exemple de code non sécurisé

```
<?php

if (isset($_SERVER['PHP_AUTH_USER']) &&
$_SERVER['PHP_AUTH_USER']==$_SERVER['PHP_AUTH_PW']) {
           $_SESSION["auth"]=1;
           $_SESSION["login"]=$_SERVER['PHP_AUTH_USER'];
}
$auth=$_SESSION["auth"];
$login=$_SESSION["login"];
# deux choses à éviter
# register_globals = On
# ET
# ce bout de code qui revient au même
foreach ( $_GET as $k => $v) {
    $$k=$v;
}
```

© ENI Editions - All rigths reserved - Algeria Educ

```
if ($auth==1) {
    echo "<H1>Bienvenue, $login</h1>";
} else {
     header('WWW-Authenticate: Basic realm="My Realm"');
    header('HTTP/1.0 401 Unauthorized');
}
?>
```

Résultat standard

Au premier appel, une popup apparaît et demande l'authentification. Le mécanisme est basique. Le mot de passe doit simplement être équivalent au login. Cependant, la présence d'un paramètre register_globals à « on » ou le code présenté dans l'exemple contournent intégralement le mécanisme d'authentification pour le rendre obsolète.

Résultat de l'attaque

http://localhost/php/secu/get.php?auth=1&login=eni

L'appel avec ces paramètres invalide toute authentification précédente et garantit l'usurpation d'identité et ceci même après une première authentification réussie.

Sécurisation du script

```
<?php

if (isset($_SERVER['PHP_AUTH_USER']) &&
$_SERVER['PHP_AUTH_USER']==$_SERVER['PHP_AUTH_PW']) {
        $_SESSION["auth"]=1;
        $_SESSION["login"]=$_SERVER['PHP_AUTH_USER'];
}
$auth=$_SESSION["auth"];
$login=$_SESSION["login"];
# deux choses à faire
# register_globals = Off
# ET
# ce bout de code qui invalide tous les paramètres passés dans
l'url à l'exception de la liste spécifiée
$params_autorises=array("debug", "verbose", "message");

foreach ( $_GET as $k => $v) {
    if (!in_array($k, $params_autorises)) {
        trigger_error("GET:$k :pas un paramètre utilisable
Valeur=[ $v ]", E_USER_WARNING);
        unset($_GET[$k]);

    }
    else $$k=$v;
}
if ($auth==1) {
    echo "<H1>Bienvenue, $login</h1>";
} else {
     header('WWW-Authenticate: Basic realm="My Realm"');
    header('HTTP/1.0 401 Unauthorized');
}
if (isset($message)) echo "<h2>$message</h2>";
echo "<pre>".print_r($_GET, true)."</pre>";
?>
```

Résultat de la seconde attaque

Ici, n'est pris en compte que ce qui est autorisé explicitement. Seuls les paramètres debug, message et verbose seront utilisés par le script. Les autres paramètres seront invalidés sur le champ afin de ne pas entraîner d'effet de bord quelconque, et ceci, même si le paramètre register_globals est positionné à Off.

http://localhost/php/secu/get_secu.php?message=%22cool%22&auth=1&login=eni

```
Bienvenue, toto
```


© ENI Editions - All rigths reserved - Algeria Educ

```
"cool"
Array
(
    [message] => "cool"
)
```

Le résultat montre bien la sécurisation complète des paramètres et l'incapacité d'usurper l'identité des utilisateurs authentifiés.

© ENI Editions - All rigths reserved - Algeria Educ

Sécurisation de l'utilisation des formulaires

1. Les faiblesses d'un formulaire

Une propriété importante du protocole HTTP est qu'il est sans état.

De ce postulat, le Web s'est enrichi des notions de session et de cookie afin de pallier ce problème ou du moins le contourner, de manière à offrir une capacité à persister l'état de certaines informations entre plusieurs appels.

Le fait d'être sans état a rendu le protocole HTTP très efficace et lui a fait connaître une expansion sans précédent ! Mais cela a entraîné un certain nombre de problèmes tels que la non-intégration de la notion de transaction entre plusieurs appels de procédures complexes.

Les formulaires sont donc aujourd'hui victimes de trois grands types d'attaque :

- L'altération des données formulaires (ajout ou modification de champ).
- L'accès au formulaire de seconde ou nième étape sans passer par les formulaires précédents.
- L'altération de destination, pour usurper des informations.

Les deux premières attaques visent essentiellement les serveurs alors que la troisième vise le client en routant les appels vers un serveur pirate, par exemple.

2. Comment sécuriser ses formulaires ?

a. Architecture des échanges avec un formulaire

La première des faiblesses d'un échange d'informations par formulaire réside dans la mauvaise conception des échanges entre le client et le serveur.

Contrôler l'ensemble des champs et leur domaine de validité systématiquement.

Comme dans le cas de l'utilisation du mode GET, il faut rejeter systématiquement les paramètres douteux et les tracer dans vos erreurs si nécessaire.

```
foreach ( $_POST as $k => $v) {
    if (!in_array($k, $params_autorises)) {
        trigger_error("POST:$k :pas un paramètre utilisable
Valeur=[ $v ]", E_USER_WARNING);
        unset($_POST[$k]);
    }
    else $$k=$v;
}
```

Cela n'améliore vraiment la sécurité que si le principe de minimisation des informations est appliqué. Ce qu'il faut comprendre par minimisation des informations échangées est que **le formulaire ne doit jamais être utilisé pour stocker de l'information entre deux appels**. Pour stocker de l'information entre deux appels, il existe plusieurs solutions telles que le stockage d'information en session, dans des fichiers temporaires, des bases de données ou des systèmes de cache.

Le cas d'école pour ce type de pratique reste le cas du site de e-commerce qui envoyait à ses clients le prix de l'article en cours de commande dans un champ caché du formulaire en pensant que rien n'était risqué dans cette pratique.

Des clients malintentionnés ont modifié la valeur de ce champ, envoyant un prix ridiculement faible.

La transaction, une fois enregistrée en base de données, a subi un traitement automatique de l'envoi, au prix fourni, sans aucune vérification.

Ce type d'attaque n'exploite ni l'injection de paramètres dans le formulaire, ni le changement de format des champs de données. Cela signifie qu'une attaque peut être réalisée en respectant totalement les règles établies pour les paramètres.

© ENI Editions - All rigths reserved - Algeria Educ

Pour se protéger intégralement de ce type de fraude, il faut donc :

- Ne pas utiliser le formulaire comme zone de stockage.
- Vérifier la cohérence des informations du formulaire.

Limitez les informations échangées au strict minimum.

b. Injection de code malicieux

Afin de limiter l'injection de code JavaScript permettant la modification du résultat affiché, il est impératif d'utiliser la validation du type de chaque donnée insérée dans votre formulaire. Définir un type pour chaque champ est important car l'utilisation gagne en qualité (vous n'avez pas saisi un nombre de jours valide, par exemple). De plus, il évite l'injection de code JavaScript. Ce code malicieux en JavaScript peut avoir deux objectifs :

- Attaque XSS visant l'utilisateur en utilisant la confiance de l'utilisateur dans le serveur Web.
- Attaque CSRF visant le serveur en utilisant la confiance du serveur dans l'utilisateur du service.

Afin de bloquer ce type d'attaque, il existe deux stratégies que vous pouvez et devez combiner :

- La vérification du type.
- La conversion des caractères en entités HTML.

c. Fonctions de validation de type PHP

Pour valider le type, il existe plusieurs fonctions PHP intégrées permettant de valider les types primaires utilisables. Toutes ces fonctions renvoient un booléen (TRUE ou FALSE).

Fonction	Description
is_array	Cette fonction détermine si l'argument est un tableau.
is_bool	Cette fonction détermine si l'argument est un booléen.
is_scalar	Cette fonction détermine si l'argument est une variable scalaire.
is_callable	Cette fonction détermine si l'argument est utilisable pour un appel de fonction PHP.
is_float, is_real, is_double	Ces fonctions déterminent si l'argument est un nombre flottant.
is_int, is_long, is_integer	Ces fonctions déterminent si l'argument est un nombre entier.
is_null	Cette fonction détermine si l'argument est nul.
is_numeric	Cette fonction détermine si l'argument est une valeur numérique (flottante ou entière).
is_object	Cette fonction détermine si l'argument est un objet PHP.
is_resource	Cette fonction détermine si l'argument est une ressource. Une ressource étant une référence vers une variable externe, comme par exemple une connexion MySQL.
is_string	Cette fonction détermine si l'argument est une chaîne de caractères.

© ENI Editions - All rigths reserved - Algeria Educ

d. Fonction de conversion des entités HTML

Il s'agit ici de la fonction permettant tout simplement d'éviter l'exécution du script.

En effet, si vous êtes pressé par les résultats et les délais de production du code et que vous n'avez pas le temps de mettre en place une validation complète des paramètres, utilisez au minimum cette fonction qui a un rôle magique et majeur pour la sécurité. Cette fonction se nomme htmlentities.

Cette fonction traduit tous les caractères spéciaux en entités HTML. Cela permet tout simplement d'invalider le code malicieux car il sera converti en une chaîne contenant des entités HTML et ne sera donc jamais exécuté.

Exemple de code non sécurisé

```
<?php

function AfficherForm() {
   echo '<div id="titre"><h2>Entrer votre message</h2><form
method="post" action="' .
htmlentities($_SERVER['PHP_SELF']).'"></div>';

    echo '<div id="form">
    <table>
        <tr>
          <td>Label:</td>
          <td><input type="text" name="label" size="30" /></td>
        </tr>
        <tr>
      <td colspan="2"><input type="submit" name="msg"
value="Submit" /></td>
        </tr>
      </table>
    </form></div>';

}


function debut() {echo "<html><head></head><body>";  }

function fin() {echo "</body></html>";   }


debut();
afficherForm();

if (isset($_POST['msg']) and isset($_POST['label'])) {
    echo '<div id="result"><pre>';
    echo $_POST['label'];
    echo '</pre></div>';

  }
fin();
?>
```

Résultat du script

Injection de code malicieux dans le champ label.

© ENI Editions - All rigths reserved - Algeria Educ

```
<script>
    document.getElementById('titre').innerHTML = "<h2><font
color='red'>ENTETE MODIFIE PAR XSS</font></h2>";
    document.getElementById('result').innerHTML = "Tout va bien
!!!!";
</script>
```

Résultat de l'injection

Ici, le code injecté est traduit et exécuté par le navigateur.

Le code HTML du titre est remplacé par un en-tête XSS rouge et le comportement normal est simulé par l'ajout d'une ligne contenant un message sous le formulaire.

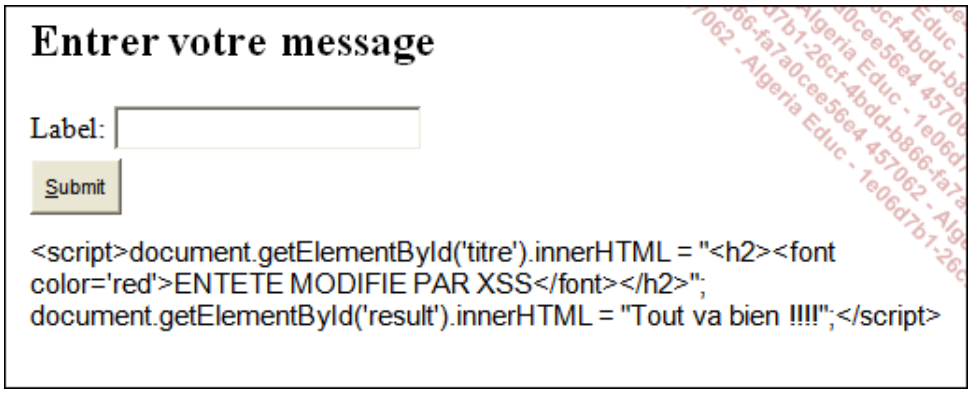
Code du formulaire protégé par la fonction htmlentities

```
<?php

function AfficherForm() {
   echo '<div id="titre"><h2>Entrer votre message</h2><form
method="post" action="' .
htmlentities($_SERVER['PHP_SELF']).'"></div>';

    echo '<div id="form">
      <table>
        <tr>
          <td>Label:</td>
          <td><input type="text" name="label" size="30" /></td>
        </tr>
        <tr>
      <td colspan="2"><input type="submit" name="msg"
value="Submit" /></td>
        </tr>
      </table>
    </form></div>';

}


function debut() {echo "<html><head></head><body>";  }

function fin() {echo "</body></html>";   }


debut();
afficherForm();

if (isset($_POST['msg']) and isset($_POST['label'])) {
    echo '<div id="result"><pre>';
    echo $_POST['label'];
    echo '</pre></div>';

  }
fin();
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

La fonction htmlentities traduisant les balises HTML en code d'entités HTML, le code JavaScript ne peut plus être traduit ni exécuté.

Résultat d'une injection malicieuse



Le code JavaScript n'est donc plus interprété et votre formulaire est protégé.

3. Exemple complet de sécurisation de formulaire

Exemple de code non sécurisé

```
<?php
$login="fournisseur1";

function AfficherForm1() {
   echo '<h2>Choisir votre produit</h2><form method="post"
action="' . htmlentities($_SERVER['PHP_SELF']).'
       ">
      <table>
        <tr>
          <td>Produit:</td>
          <td><input type="text" name="label" size="30" /></td>
        </tr>

        <tr>
          <td>Quantité:</td>
          <td><input type="text" name="qte" size="3" /></td>
        </tr>

        <tr>
          <td colspan="2"><input type="submit" name="subf1"
value="Submit" /></td>
        </tr>
      </table>
    </form>';

}

function AfficherForm2($label, $quantite, $prix, $login) {
   echo '<h2>Valider votre produit</h2><form method="post"
action="' . htmlentities($_SERVER['PHP_SELF']).'
       ">
      <table>
        <tr>
          <td>Produit:</td>
          <td><input type="text" name="label" size="30"
value="'.$label.'"/></td>
        </tr>

        <tr>
```

© ENI Editions - All rigths reserved - Algeria Educ

```
          <td>Quantité:</td>
          <td><input type="text" name="qte" size="3"
value="’.$quantite.’"/></td>
        </tr>

        <tr>
          <td colspan="2"><input type="submit" name="subf2"
value="Submit" /></td>
        </tr>
      </table>
      <input type="hidden" name="prixUnitaire"
value="’.$prix.’"/>
      <input type="hidden" name="login"  value="’.$login.’"/>
    </form>’;

}

function debut() {echo "<html><head></head><body>";  }

function fin() {echo "</body></html>";   }
  if (!isset($_POST[’subf1’]) and !isset($_POST[’subf2’])) {
    debut();
    afficherForm1();
    fin();
    exit(0);
  }
  if (isset($_POST[’subf1’])) {
    debut();

    // POST submission, validate input
    if (trim($_POST[’label’]) == ’’) {
      die(’ERREUR: Pas de label’);
    }
    if (trim($_POST[’qte’]) == ’’) {
      die(’ERREUR: Pas de quantité’);
    }
    afficherForm2($_POST[’label’], $_POST[’qte’], "10", $login);
    fin();
    exit(0);
  }
  if (isset($_POST[’subf2’])) {
      debut();
      echo "<H2>JE VALIDE:</H2><pre>";
      echo "\n * Login:". $_POST[’login’];
      echo "\n * Produit:". $_POST[’label’];
      echo "\n * Quantité:". $_POST[’qte’];
      echo "\n * prixUnitaire:". $_POST[’prixUnitaire’];
      echo "\n * prix Total:". $_POST[’prixUnitaire’]*$_POST[’qte’];
      echo "</pre>";
      fin();
  }
?>
```

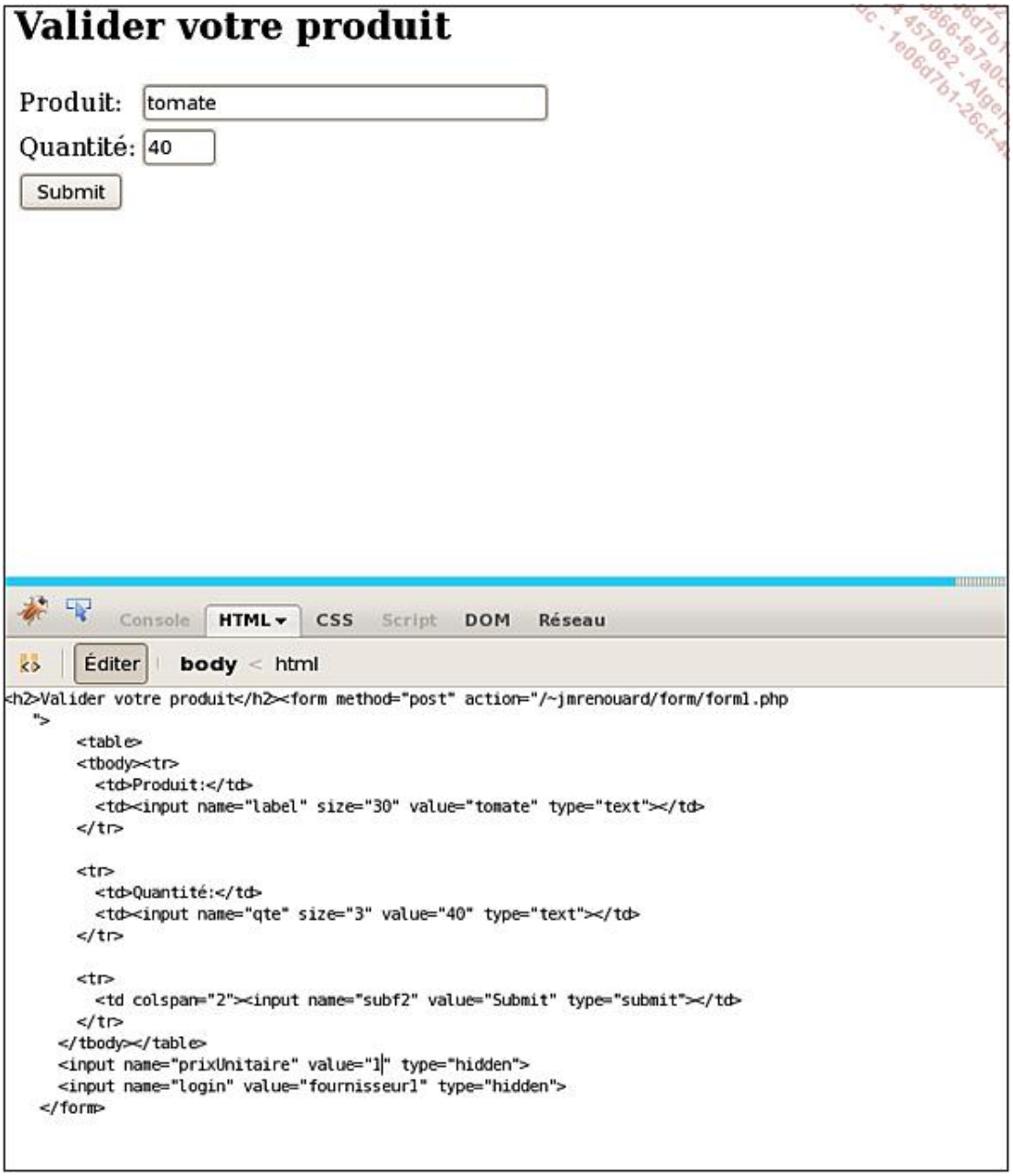
Résultat standard

© ENI Editions - All rigths reserved - Algeria Educ

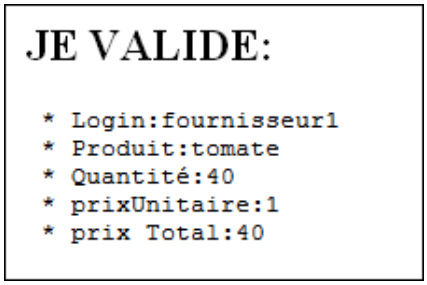
Résultat de l'attaque

Avec l'outil Firebug, il est possible d'éditer directement le formulaire avec l'éditeur de code HTML et de modifier directement depuis votre navigateur la valeur des champs cachés (Hidden) du formulaire de validation.

© ENI Editions - All rigths reserved - Algeria Educ



Et par magie, le prixUnitaire a été divisé par 10 et la commande est validée !



Exemple de code sécurisé

```
<?php
session_start();
$login="fournisseur1";

$tabPrix['patate']="50";
$tabPrix['tomate']="60";
```


© ENI Editions - All rigths reserved - Algeria Educ

```
$tabPrix['radis']="20";

function AfficherForm1() {
   echo '<h2>Choisir votre produit</h2><form method="post"
action="' . htmlentities($_SERVER['PHP_SELF']).'
       ">
      <table>
        <tr>
          <td>Produit:</td>
          <td><input type="text" name="label" size="30" /></td>
        </tr>

        <tr>
          <td>Quantité:</td>
          <td><input type="text" name="qte" size="3" /></td>
        </tr>

        <tr>
          <td colspan="2"><input type="submit" name="subf1"
value="Submit" /></td>
        </tr>
      </table>
    </form>';

}

function AfficherForm2($label, $quantite) {
   echo '<h2>Valider votre produit</h2><form method="post"
action="' . htmlentities($_SERVER['PHP_SELF']).'
       ">
      <table>
        <tr>
          <td>Produit:</td>
          <td><input type="text" name="label" size="30"
value="'.$label.'"/></td>
        </tr>

        <tr>
          <td>Quantité:</td>
          <td><input type="text" name="qte" size="3"
value="'.$quantite.'"/></td>
        </tr>

        <tr>
          <td colspan="2"><input type="submit" name="subf2"
value="Submit" /></td>
        </tr>
      </table>
    </form>';

}

function debut() {echo "<html><head></head><body>";  }

function fin() {echo "</body></html>";   }

if (isset($_POST['label']) and
!array_key_exists($_POST['label'],$tabPrix)) {
    debut();
        echo '<font color="red">des '.$_POST['label'].'s ne sont
pas dans nos produits.</font>';
    afficherForm1();
    fin();
    exit(0);

}
if (!isset($_POST['subf1']) and !isset($_POST['subf2'])) {
    debut();
    afficherForm1();
```

© ENI Editions - All rigths reserved - Algeria Educ

```
    fin();
    exit(0);
  }
  if (isset($_POST['subf1'])) {
    debut();

    // POST submission, validate input
    if (trim($_POST['label']) == '') {
      die('ERREUR: Pas de label');
    }
    if (trim($_POST['qte']) == '') {
      die('ERREUR: Pas de quantité');
    }
    afficherForm2($_POST['label'], $_POST['qte']);
    $_SESSION["label"]=$_POST['label'];
    $_SESSION["prixUnitaire"]=$tabPrix[$_SESSION["label"]];
    $_SESSION["label"]=$_POST['label'];
    $_SESSION["login"]=$login;
    fin();
    exit(0);
  }
  if (isset($_POST['subf2'])) {
    if ($_POST['label']!=$_SESSION['label']) {
          debut();
          echo '<font color="red">des '.$_POST['label'].'s ne sont
pas des '.$_SESSION['label'].'s</font>';
          $_SESSION["label"]=$_POST['label'];
          $_SESSION["prixUnitaire"]=$tabPrix[$_SESSION["label"]];

          afficherform2($_POST['label'], $_POST['qte']);
          fin();
          exit(0);
      }

      debut();
      echo "<H2>JE VALIDE:</H2><pre>";
      echo "\n * Login:". $_SESSION['login'];
      echo "\n * Produit:". $_POST['label'];
      echo "\n * Quantité:". $_POST['qte'];
      echo "\n * prixUnitaire:". $_SESSION['prixUnitaire'];
      echo "\n * prix Total:".
$_SESSION['prixUnitaire']*$_POST['qte'];
      echo "</pre>";
      fin();
  }
?>
```

Résultat standard

Le fonctionnement est le même que dans le script précédent !

Résultat de l'attaque

Le script empêche directement le choix de produits inexistants !

Cela permet de garantir d'éventuelles erreurs de frappe.

© ENI Editions - All rigths reserved - Algeria Educ

des rrs ne sont pas dans nos produits.

Choisir votre produit

Produit:

Quantité:

Submit

Il n'est plus possible de changer le produit et garder le prix initial du produit choisi.
des patates ne sont pas des tomates

Valider votre produit

Produit: patate

Quantité: 20

Submit

Lors de la validation, le prix unitaire est celui qui a été validé après avertissement du changement du produit.

JE VALIDE:

* Login:fournisseur1
* Produit:patate
* Quantité:20
* prixUnitaire:50
* prix Total:1000

4. Composant pour les formulaires

Afin de garantir une utilisation systématique des validations, il suffit de choisir des composants dédiés à la gestion des formulaires.

Voici donc un exemple utilisant le composant PEAR QuickForm2 permettant de faciliter et donc de favoriser l'utilisation systématique des validations. Ce composant permet de simplifier le code et sa lisibilité.

a. Installation du module PEAR HTML_QuickForm2

Nous exécutons le programme PEAR fourni avec PHP afin d'installer un composant fort utile pour simplifier la création, la génération et la validation de nos formulaires.

```
#  pear install "channel://pear.php.net/HTML_Common2-2.0.0RC1"
WARNING: channel "pear.php.net" has updated its protocols, use
"pear channel-upd
ate pear.php.net" to update
downloading HTML_Common2-2.0.0RC1.tgz ...
Starting to download HTML_Common2-2.0.0RC1.tgz (7,598 bytes)
.....done: 7,598 bytes
```

© ENI Editions - All rigths reserved - Algeria Educ

```
install ok: channel://pear.php.net/HTML_Common2-2.0.0RC1

# pear install "channel://pear.php.net/HTML_QuickForm2-0.4.0"
WARNING: channel "pear.php.net" has updated its protocols, use
"pear channel-update pear.php.net" to update
downloading HTML_QuickForm2-0.4.0.tgz ...
Starting to download HTML_QuickForm2-0.4.0.tgz (101,758 bytes)
.......................done: 101,758 bytes
install ok: channel://pear.php.net/HTML_QuickForm2-0.4.0
```

b. Test d'installation du module Pear HTML_QuickForm2

```
< ?php
require_once 'HTML/QuickForm2.php';
$form = new HTML_QuickForm2('monformulaire');
?>
```

Si rien n'apparaît dans les traces, alors l'installation s'est bien passée !

Vous pouvez obtenir ce même résultat en ligne de commande.

L'appel en ligne de commande de l'inclusion d'un module inexistant provoque une traîne de messages d'alerte.

```
$ php -r "require_once 'HTML/QuickForm365.php';"
PHP Warning:  require_once(HTML/QuickForm365.php): failed to open
stream: No such file or directory in Command line code on line 1

Warning: require_once(HTML/QuickForm365.php): failed to open
stream: No such fil
e or directory in Command line code on line 1
PHP Fatal error:  require_once(): Failed opening required
'HTML/QuickForm365.php
' (include_path='.:/usr/share/pear:..') in Command line code on
lin
e 1

Fatal error: require_once(): Failed opening required
'HTML/QuickForm365.php' (include_path='.:/usr/share/pear:..') in
Command line code on line 1
```

Après l'installation, le fonctionnement de notre module PHP est parfaitement normal et ne provoque plus de message d'erreur.

```
$ php -r "require_once 'HTML/QuickForm2.php';"
$
```


© ENI Editions - All rigths reserved - Algeria Educ

Sécurisation de l'accès aux sessions PHP

Il est parfois possible pour un attaquant de modifier son identifiant de session afin de récupérer les informations concernant un autre utilisateur. Voila pourquoi de nombreux experts indiquent que pour réaliser une attaque, le pirate devient membre ou utilisateur authentique puis cherche à élargir ses droits ou à usurper ceux des autres utilisateurs.

Dès que vous avez accès au service, il vous suffit de tenter de modifier votre identifiant de session. Dès que vous avez intercepté une autre session, vous pouvez vous faire entièrement passer pour autrui.

Attention, je ne vous dis pas de le faire ni que cela est trivial et rapide à mettre en place, cependant, avec un peu de travail, il est possible de récupérer les identifiants de session.

La première des stratégies consiste à exécuter un script de listing de l'ensemble des sessions. Vous pourrez obtenir cette information si vous avez la possibilité d'envoyer votre propre script sur le serveur et de l'exécuter.

Voici un code simple pour récupérer les données des sessions dont le principe consiste à balayer le répertoire par défaut des sessions PHP, et de construire une URL permettant de démarrer la session avec l'identifiant de session repéré dans le répertoire.

Il s'agit d'un script dérivé permettant d'explorer les fichiers du système d'exploitation en balayant les répertoires de configuration afin d'y découvrir un fichier de configuration rempli de logins et de mots de passe.

```
<?php

if (isset($_GET['PHPSESSID'])) {
    session_start();
    print "<pre>".print_r( $_SESSION, true)."</pre>";
}
echo "<hr/>";
$rep_session="C:/wamp/tmp";

if ($handle = opendir($rep_session)) {
    echo "<ul>";
    /* Ceci est la façon correcte de traverser un dossier. */
    while (false !== ($file = readdir($handle))) {
        if ($file == '.' or $file =='..') continue;
        $sessionid=substr($file, 5);
        echo "<li><a
href='?PHPSESSID=$sessionid'>$sessionid</a></li>\n";
    }
    echo "</ul>";

    closedir($handle);
}
?>
```

Ma préférence va de loin à l'usurpation des sessions PHP des outils d'administration (comme PhpMyAdmin) qui sont, par défaut, remplis de mots de passe stockés en session.

Exemple affichant les informations de la session

```
Array
(
 [cms_admin_username] => on
 [login_user_username] => on
 [cmsuserkey] => 062f4ca9
 [cms_admin_user_id] => 1
)
```

- 08d2crdktgd9njgupmusibbja3
- 2jjgefsdv462qvqrkcer24iug3
- 42htd1dgttrs5srin904cpn9t3
- 4ttjpflja0klkteqoskgd47spr1ifkeu

© ENI Editions - All rigths reserved - Algeria Educ

- 8fd864e38ti1epcda6gbem7fg6
- bl26ifra8ghibb8rttcjhd7st0
- cfh67eht61ebcroe1kqe3ci6iunkp5ep
- h83uh30j211tah7vg662ahbb65
- hjvqc2jploav8j2k6k7c8uosl4
- js9s04et8t68b4o9bskuhppso6
- o9mhk37pv4qkfql78ouq75u9g9vm28ts
- oh0qenl2rpphrihsme19holud5
- otieab89tmj8j9mtf69uo1o1u6
- phj22bici8dv8e1bcha0mi1l02
- qeg9sfqari7q31tv335v086t50bmrfco
- s8afg2imln13f8t6vkie9fcvc7
- vbgm79a973cm7j4ls1np0up6qu4ij2bo

Pour se prémunir d'une telle utilisation des scripts PHP, voici quatre techniques de base permettant de contourner ce type de vol d'information. Les trois premières tentent de juguler les accès inappropriés aux fichiers et donc aux données des sessions. La dernière propose une configuration invalidant la possibilité de passer par des URL contenant le numéro de session.

1. Modification de la technique de sauvegarde des sessions

La première technique consiste à stocker vos informations en dehors du répertoire par défaut, dans une base de données par exemple. Deux exemples de code, complets, sont présents dans le chapitre sur la haute disponibilité PHP et la gestion des sessions partagées.

2. Limiter l'accès aux répertoires par Apache

Un second exemple de contournement est de réduire les droits d'accès au système de fichiers. Pour cela une directive pour le serveur Apache consiste à définir un RootDocument extrêmement restreint.

```
DocumentRoot /var/www/html/session
```

La directive RootDocument d'Apache offre la possibilité de redéfinir le répertoire Racine pour l'ensemble des accès ou par hôte virtuel, par Ip ou par nom de domaine.

3. Réduire les droits d'accès au niveau du système d'exploitation

Un troisième exemple consiste à restreindre fortement l'accès à l'utilisateur Apache sur votre arborescence de fichiers afin qu'il ne puisse pas lire tous les fichiers ni les répertoires sensibles. Cette solution reste cependant limitée car l'écriture des informations de session doit avoir lieu et, si l'utilisateur ne peut réaliser les opérations d'écriture, alors il n'est pas possible de récupérer les sessions. Quel dommage !

4. Bannir le passage de session par URL

© ENI Editions - All rigths reserved - Algeria Educ

La quatrième technique consiste à désactiver le passage de session par l'URL.

Dans le fichier php.ini, il suffit de positionner les deux paramètres suivants :

```
session.use_trans_sid=0
session.use_only_cookies=1
```

À noter que cette technique ne permet pas de limiter la possibilité de forger des cookies de session côté client. Cependant, il empêche de faire le lien fichier/numéro de session.

En effet, dans ce type d'attaque, le plus difficile consiste à trouver une session valide. Si votre script ne le permet plus, alors l'attaque devient plus complexe et donc limitée à un nombre d'individus encore plus restreint.

5. Régénération d'identifiant de session

La cinquième technique consiste à interdire la régénération d'identifiant de session quand celle-ci est déjà créée. L'idée est de régénérer un identifiant de session si la session n'est pas active. Dans le cas contraire, il est possible de récupérer toutes les informations de la session.

6. Validation du navigateur client

L'idée est de vérifier par code PHP que le client est bien le même que celui qui a initialisé la session. S'il est possible de récupérer la session par script ou par lien spécifique, il est plus difficile de connaître à la fois le mécanisme de génération d'une somme de contrôle basée sur l'en-tête HTTP User-Agent du navigateur et d'une clé aléatoire générée initialement.

Le User-Agent est la chaîne décrivant le navigateur, sa version et toutes les informations utiles permettant l'identification du navigateur utilisé.

7. Combinaison des deux solutions pour sécuriser les sessions

L'idée est de forcer la régénération de session et de créer une somme de contrôle pour chaque session afin de garantir la relation navigateur/session. L'usurpation ne sera plus liée à la connaissance de l'identifiant de session mais à d'autres facteurs très difficiles à trouver tels que : le code de génération de la somme clé, la présence de données aléatoires et l'utilisation de l'en-tête HTTP User-Agent identifiant spécifiquement le navigateur du client d'origine.

Il s'agit donc d'une protection forte, très difficile à contourner sans avoir accès physiquement à la machine client de l'utilisateur initial.

Code de création d'une session

```
<?php

session_start();

function chaineAleatoire(int $taille=16) {

    $base='ABCDEFGHKLMNOPQRSTWXYZabcdefghjkmnpqrstwxyz123456789';
    $max=strlen($base)-1;
    $result='';
    while (strlen($result)<$taille+1)
$result.=$base{mt_rand(0,$max)};

    return $result;
}

if (!isset($_SESSION['init']))
{
    session_regenerate_id();
    $_SESSION['init'] = true;
    $_SESSION['ALEATOIRE']=chaineAleatoire(16);
    $_SESSION['HTTP_USER_AGENT'] =
```

© ENI Editions - All rigths reserved - Algeria Educ

```
md5($_SERVER['HTTP_USER_AGENT'].$_SESSION['ALEATOIRE']);
}

if ($_SESSION['HTTP_USER_AGENT'] !=
md5($_SERVER['HTTP_USER_AGENT'].$_SESSION['ALEATOIRE']))
{
  exit;
}


?>
```


© ENI Editions - All rigths reserved - Algeria Educ

Sécurisation de l’accès aux bases de données

Les bases de données souffrent d’attaques au travers du code PHP :

- La destruction ou l’altération de données.
- Le vol de données.

La recette de base est toujours la même : on injecte dans un champ des données particulières et, dès que ces informations sont transmises à la base de données, alors une requête est exécutée.

Exemple :

```
SELECT * FROM UTLISATEUR WHERE LOGIN='eni' ;
```

Le paramètre de la requête SQL, « eni » est en fait le contenu d’un champ de saisie dans un formulaire.

Une attaque simple consiste à réécrire la requête SQL de la manière suivante :

```
SELECT * FROM UTLISATEUR WHERE LOGIN=" OR '1' = '1';
```

ou bien

```
SELECT * FROM UTLISATEUR WHERE LOGIN=" OR LOGIN LIKE '%';
```

L’attaquant passe donc la chaîne suivante en paramètre et provoque la récupération de l’intégralité du contenu de la table UTLISATEUR.

Il suffit pour cela d’injecter l’un ou l’autre des contenus suivants dans le champ de saisie Login :

```
' OR '1' = '1
```

ou bien

```
' OR LOGIN LIKE '%
```

Pour se protéger des attaques, comme celle vue dans le cas des formulaires, rien de tel que d’utiliser une fonction permettant de neutraliser l’ensemble des caractères douteux (en évaluant les fins de champs par exemple).

Cette fonction magique, c’est mysql_escape_string. Elle permet de se protéger de tous les caractères spéciaux SQL.

Code SQL de la base

```
create database utilisateur;
use utilisateur;
GRANT ALL PRIVILEGES ON utilisateur.* TO util@localhost identified
by 'util';
FLUSH PRIVILEGES;
CREATE TABLE UTILISATEUR ( ID BIGINT PRIMARY KEY, LOGIN
VARCHAR(255), NOM VARCHAR(355), PRENOM VARCHAR(255), EMAIL
VARCHAR(255));
INSERT INTO UTILISATEUR VALUES( NULL, 'admin', 'Administrateur',
'Administrateur', 'admin@service.org');
INSERT INTO UTILISATEUR VALUES( 0, 'admin', 'Administrateur',
'Administrateur', 'admin@service.org');
INSERT INTO UTILISATEUR VALUES( 1, 'jmrenouard', 'Jean-Marie',
'Renouard', 'jmrenoaurd@123solution.fr');
INSERT INTO UTILISATEUR VALUES( 2, 'amrenouard', 'Anne-Marie',
'Renouard', 'admin@123solution.fr');
```

Exemple de code vulnérable

```
<?php

function AfficherForm() {
   echo '<div id="titre"><h2>Recherche utilisateur</h2><form
```

© ENI Editions - All rigths reserved - Algeria Educ

```
method="post" action="' .
htmlentities($_SERVER['PHP_SELF']).'"></div>';

    echo '<div id="form">
    <table>
        <tr>
          <td>Login:</td>
          <td><input type="text" name="label" size="30" /></td>
        </tr>
        <tr>
      <td colspan="2"><input type="submit" name="msg"
value="Submit" /></td>
        </tr>
      </table>
    </form></div>';

}


function debut() {echo "<html><head></head><body>";  }

function fin() {echo "</body></html>";   }


debut();
afficherForm();


function getUtilisateurByLogin($login='admin') {

    $link = mysql_connect('localhost', 'util', 'util');
    $db_selected = mysql_select_db('utilisateur', $link);
    $ret=array();

    if (!$link) {
            die('Connexion impossible : ' . mysql_error());
    }

    $req = sprintf("SELECT * FROM UTILISATEUR WHERE login = '%s'",
$login);
    $result = mysql_query($req);

    echo 'Requête complète : ' . $req;
    if (!$result) {
            $message  = 'Requête invalide : ' . mysql_error()
. "\n";
            $message .= 'Requête complète : ' . $req;
            die($message);
    }

    while($ligne = mysql_fetch_assoc($result)) {
        array_push($ret, $ligne);
    }

    mysql_free_result($result);
    mysql_close($link);
    return $ret;
}

if (isset($_POST['msg']) and isset($_POST['label'])) {

    echo '<div id="result"><pre>';
    print_r( getUtilisateurByLogin($_POST['label']) );
    echo '</pre></div>';

  }
fin();
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

Résultat de l'injection SQL

```
Requête complète : SELECT * FROM UTILISATEUR WHERE login = 'admin'
OR '1' ='1'Array
(
    [0] => Array
        (
            [ID] => 0
            [LOGIN] => admin
            [NOM] => Administrateur
            [PRENOM] => Administrateur
            [EMAIL] => admin@service.org
        )

    [1] => Array
        (
            [ID] => 1
            [LOGIN] => jmrenouard
            [NOM] => Renouard
            [PRENOM] => Jean-Marie
            [EMAIL] => jmrenouard@123solution.fr
        )

    [2] => Array
        (
            [ID] => 2
            [LOGIN] => amrenouard
            [NOM] => Renouard
            [PRENOM] => Anne-Marie
            [EMAIL] => admin@123solution.fr
        )

)
```

Exemple de code protégé

Il suffit de modifier le code de la fonction getUtilisateurByLogin et d'invoquer mysql_escape_string sur chaque paramètre utilisé.

```
function getUtilisateurByLogin($login='admin') {

    $link = mysql_connect('localhost', 'util', 'util');
    $db_selected = mysql_select_db('utilisateur', $link);
    $ret=array();

    if (!$link) {
            die('Connexion impossible : ' . mysql_error());
    }

    $req = sprintf("SELECT * FROM UTILISATEUR WHERE login = '%s'",
mysql_escape_string($login));
    $result = mysql_query($req);

    echo 'Requête complète : ' . $req;
    if (!$result) {
            $message  = 'Requête invalide : ' . mysql_error()
. "\n";
            $message .= 'Requête complète : ' . $req;
            die($message);
    }

    while($ligne = mysql_fetch_assoc($result)) {
        array_push($ret, $ligne);
    }

    mysql_free_result($result);
    mysql_close($link);
    return $ret;
```

© ENI Editions - All rigths reserved - Algeria Educ

```
}
```

Résultat de la tentative d'injection SQL

```
Requête complète : SELECT * FROM UTILISATEUR WHERE login =
'admin\' OR \'1\' =\'1'

Array
(
)
```


© ENI Editions - All rigths reserved - Algeria Educ

Optimisation Apache en fonction de votre trafic

Apache possède de nombreuses fonctionnalités permettant d'améliorer les performances globales de votre serveur.

Il existe plusieurs points d'amélioration basés, pour l'essentiel, sur un principe général de la réduction du volume de données échangées entre parties.

Cela peut se traduire par :

- La mise en place de caches client par les en-têtes de cache.
- La mise en place de caches des ressources statiques côté serveur (images, JavaScript, css, etc.)
- La compression des transferts de données.
- La mise en place de caches de résultats d'exécution des scripts PHP.
- La mise en place de caches de compilation des scripts PHP.
- La mise en place de caches partiels de données.
- La mise en place de caches de données pour l'accès aux bases de données.

© ENI Editions - All rigths reserved - Algeria Educ

Optimisation des caches navigateur et en-tête HTTP

Il est possible d'optimiser ou de garantir le rafraîchissement de chacune de vos pages à chaque appel.

Si vous ne voulez pas permettre un fonctionnement parasitaire de votre application Web au travers des relais Web de certaines entreprises, il est impératif de mettre en place des en-têtes HTTP spécifiques afin que votre page HTML générée par votre script PHP ne soit pas mise «sauvagement » dans le cache Web d'un proxy au modèle d'optimisation trop poussé.

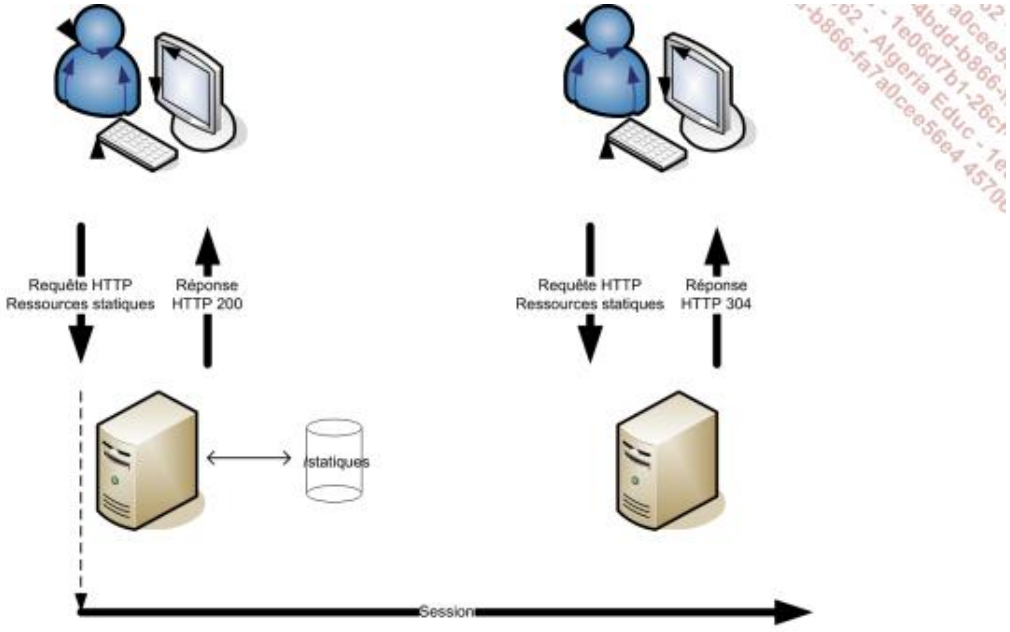
Cela signifie que si vous ne souhaitez pas voir le résultat de votre script mis une fois pour toutes en cache, il est impératif que vous positionniez le code PHP suivant au début de votre code.

```
<?php
 header('Expires: Mon, 26 Jul 1997 05:00:00 GMT');
 header('Cache-Control: no-store, no-cache, must-revalidate');
 header('Cache-Control: post-check=0, pre-check=0', FALSE);
 header('Pragma: no-cache');
?>
```

Les directives précédentes positionnent la date d'expiration de votre page à une date antérieure à la date courante.

Les directives de cache des seconde et troisième lignes sont des directives pour la version HTTP 1.1 (c'est-à-dire la plus récente) et indiquent aussi que l'on ne souhaite pas stocker, ni réaliser de mise en cache et que tout contenu doit être revalidé. Aucune validation ne doit être réalisée avant ni après chaque requête.

La quatrième ligne représente une direction pour la version 1.0, indiquant que l'on ne souhaite pas gérer un cache de page côté navigateur.



Fonctionnement vis-à-vis du contenu statique
avec gestion des en-têtes et du contenu avec des données d'expiration

Le premier appel produit une récupération de la ressource par transfert HTTP. Cependant, dès le second appel, le client transmet les en-têtes descriptifs de la ressource dont Etag, la date de dernière modification etc. et le serveur peut, s'il considère que la ressource est à jour sur le navigateur client, renvoyer un code HTTP « 304-Not Modified » indiquant simplement que la version est à jour. Le serveur, dans ce cas, ne retransmet pas les données de la ressource, garantissant un échange optimal.

Vous pouvez comprendre maintenant pourquoi la première connexion sur de nombreux sites est plus lente au premier appel et que ce temps de réponse chute grandement dès le second clic de souris.

© ENI Editions - All rigths reserved - Algeria Educ

Cache serveur Apache : mod_cache

Apache offre de nombreux modules dont un module de gestion des ressources statiques dans un cache mémoire.

Le cache fonctionne selon les principes suivants :

- Ce qui n'est pas en cache doit être mis en cache.
- Si la ressource est disponible en cache, il faut l'utiliser.
- Pas de redistribution si le cache client est à jour.

La ressource est récupérée sur le disque ou, depuis un autre serveur quand le serveur fonctionne en tant que relais HTTP. Dès qu'elle est récupérée, la ressource est mise en cache, soit sur le disque, soit en mémoire. Les appels suivants produisent une réponse directement depuis les données du cache sans avoir à rechercher les ressources réelles.

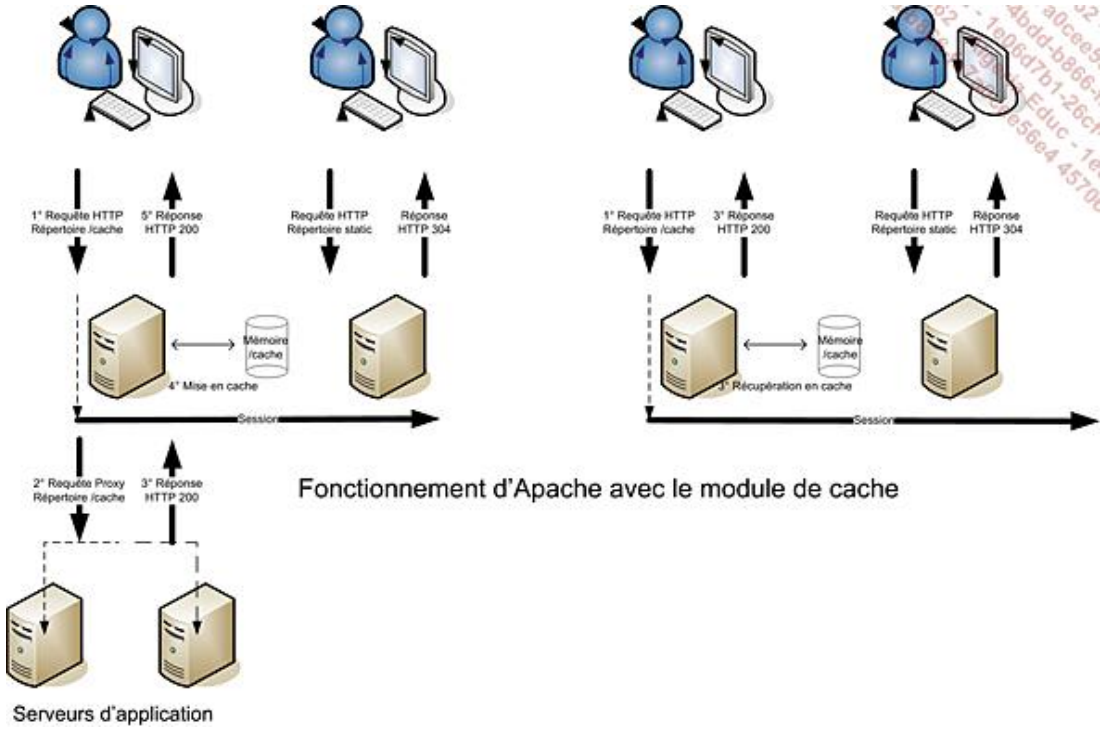
Exemple de paramétrage

Tout ce qui passe dans le répertoire/cache est automatiquement mis en cache mémoire, à moins que sa taille soit inférieure à environ 1 ko ou supérieure à environ 1 Mo. Le cache aura une taille d'environ 95 Mo.

```
CacheEnable mem /cache
MCacheSize 100000000
MCacheMaxObjectCount 1000
MCacheMinObjectSize 100
MCacheMaxObjectSize 1000000
```

- CacheEnable : identification du type de mémoire et du chemin concerné par la mise en cache.
- MCacheSize : la taille globale du cache en octets.
- MCacheMaxObjectCount : nombre maximum d'URL distinctes à mettre en cache.
- MCacheMinObjectSize : taille minimum en octets d'une URL pour qu'elle soit éligible au cache.
- MCacheMaxObjectSize: taille maximum en octets d'une URL pour qu'elle soit éligible au cache.

© ENI Editions - All rigths reserved - Algeria Educ



Le premier appel met la ressource en cache mémoire, puis le premier client fonctionnera de manière optimale à partir du cache client. En effet, le cache client reste la meilleure optimisation possible du volume d'échange entre le navigateur et le serveur HTTP.

Les appels suivants produisent une utilisation directe du cache Apache sans aucune opération supplémentaire.

© ENI Editions - All rigths reserved - Algeria Educ

Compression des ressources

Une autre stratégie permettant de limiter le nombre d'informations transmises entre le client et le serveur consiste à compresser les données avant la transmission réduisant le volume d'information échangée. Ceci implique une tâche supplémentaire de compression à chaque émission et une tâche de décompression à chaque réception. Cela implique que plusieurs paramètres entrent en compte :

- La taille de la ressource.
- L'algorithme de compression.
- Le paramétrage de la compression.
- L'existence d'une solution alternative : le mode sans compression.

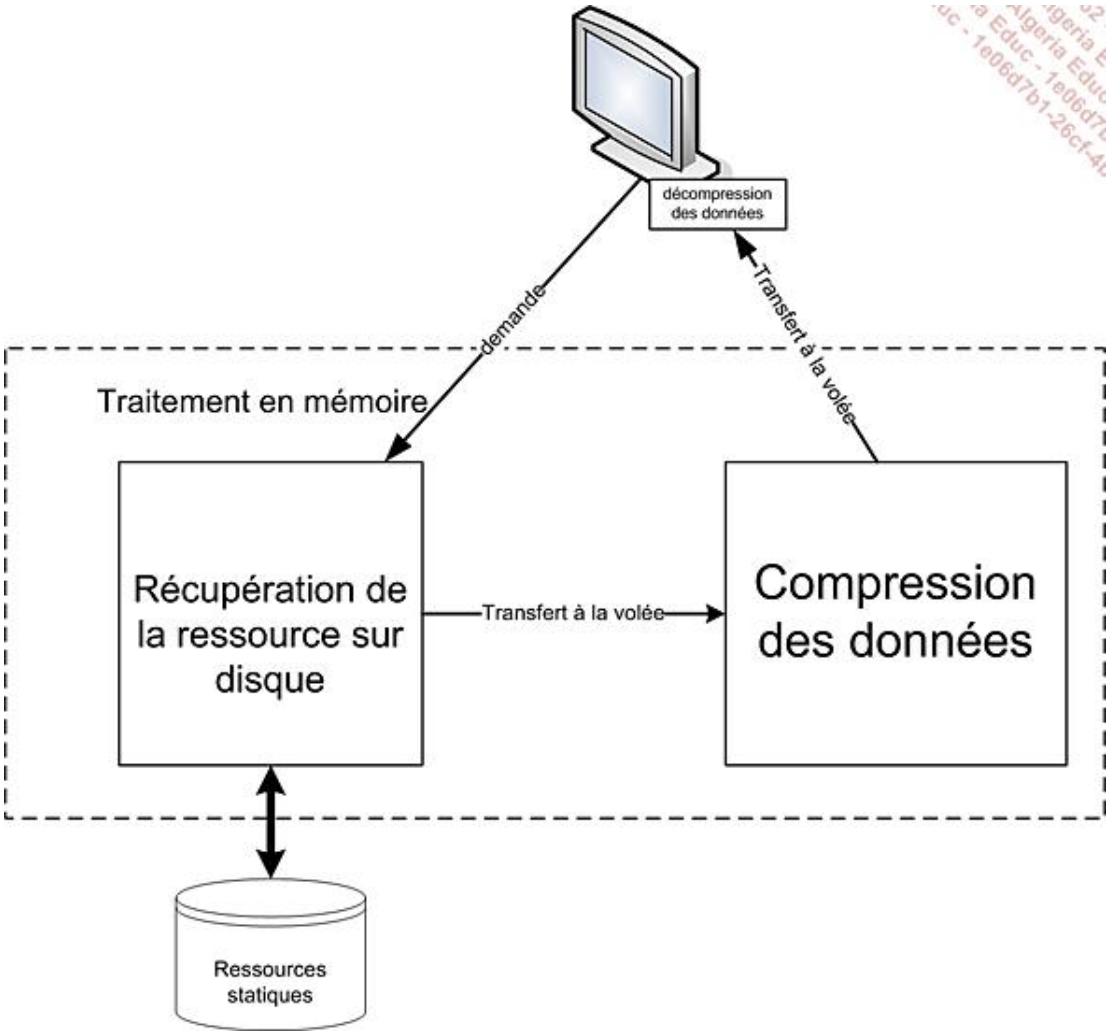
Le mécanisme actuel de compression est particulièrement efficace et passe par une implémentation dite de « compression à la volée ».

Ce mécanisme consiste à lire la ressource sur le disque dur ou sur le support de persistance et de compresser les données directement en mémoire sans jamais stocker de résultats ou de données intermédiaires.

Par ce mécanisme, le processus de compression s'insère dans une chaîne logicielle permettant de limiter l'impact de la compression, c'est-à-dire que le temps nécessaire pour fournir plusieurs ressources compressées est quasi équivalent au temps de réponse sans compression.

Cela provient du fait que l'on chaîne les opérations de compression et que cette opération est effectuée au fil de l'eau, c'est-à-dire que la phase de compression commence avant même que la phase de récupération sur le disque ne soit terminée.

Le client peut même commencer la décompression avant la fin de la récupération du fichier si celui-ci est très gros.



© ENI Editions - All rigths reserved - Algeria Educ

Paramétrage type pour la compression

```
# chargement du module
LoadModule deflate_module modules/mod_deflate.so

<IfModule mod_deflate.c>
        # activation du filtre DEFLATE
        SetOutputFilter DEFLATE

        # selon le user agent, on active ou pas
        BrowserMatch ^Mozilla/4 gzip-only-text/html
        BrowserMatch ^Mozilla/4\.0[678] no-gzip
        BrowserMatch \bMSIE\s7 !no-gzip !gzip-only-text/html

        # on exclut les types de fichier suivants :
        SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png)$ no-gzip
dont-vary
        SetEnvIfNoCase Request_URI \.(?:exe|t?gz|zip|bz2|sit|rar)$
no-gzip dont-vary
        SetEnvIfNoCase Request_URI \.pdf$ no-gzip dont-vary
        SetEnvIfNoCase Request_URI \.swf$ no-gzip dont-vary

        # niveau de compression
        DeflateCompressionLevel 1
        DeflateWindowSize 15
        DeflateMemLevel 9
        DeflateBufferSize 16192

        # indique aux proxys qu'il ne faut envoyer des réponses
en cache qu'aux clients
        # ayant envoyé l'en-tête Accept-Encoding approprié.
        # cela évite que le proxy n'envoie du contenu compressé à un
client ne le supportant pas
        <IfModule mod_headers.c>
              Header append Vary User-Agent
        </IfModule>

        # traces spécifiques du mod deflate
        DeflateFilterNote Output output_info
       # DeflateFilterNote Ratio ratio_info
        LogFormat '"%r" %b "%{User-agent}i" %{output_info}n
(%{ratio_info}n%%)' deflate
        CustomLog logs/deflate_log deflate
</IfModule>
```

Comme nous pouvons le constater, le mod_deflate possède de nombreuses possibilités et il faut donc être prudent concernant l'envoi de données compressées vers certains navigateurs ou certains équipements comme les serveurs de relais HTTP.

La directive d'activation de la compression des données :

```
SetOutputFilter DEFLATE
```

Notre exemple nous montre comment bannir certains navigateurs trop vieux ou trop spécifiques des serveurs gérant des données compressées. Comme par exemple de vieilles versions de Mozilla :

```
BrowserMatch ^Mozilla/4\.0[678] no-gzip
```

Notre exemple met aussi en évidence comment éviter la compression de données ayant déjà a priori un format de données compressé et éviter le passage par la case compression qui serait inutile, coûteuse et très faiblement efficace.

```
SetEnvIfNoCase Request_URI \.(?:exe|t?gz|zip|bz2|sit|rar)$ no-gzip
dont-vary
```


© ENI Editions - All rigths reserved - Algeria Educ

Mécanisme de cache de résultat côté serveur

L'idée du mécanisme de cache de résultat côté serveur consiste à mettre en place une zone de stockage temporaire permettant de récupérer les données sans avoir à exécuter le code source PHP permettant de créer le contenu. Ceci évite une exécution et des recherches d'information à chaque appel.

Sur une ressource dite critique, ce mécanisme permettra de prévenir l'exécution de plusieurs milliers de fois « le même script qui génère à chaque fois la même chose ».

Nous avons vu plus haut comment mettre en place un cache Apache de qualité. Cependant, il est parfois nécessaire d'effectuer des opérations de vérification avant de renvoyer une page pré calculée, afin de garantir la fraîcheur de l'information. De même, le cache Apache ne permet pas de mettre en cache une sous-partie de la ressource. De fait, il est impossible d'implémenter un mécanisme de cache sophistiqué et efficace.

Le module mod_cache de Apache est indispensable pour garantir le fonctionnement et la livraison optimale des ressources statiques telles que les images, les feuilles de style, les scripts JavaScript. Cependant, il ne sera vraisemblablement pas suffisant pour l'optimisation de vos services.

La mise en place d'un cache applicatif côté serveur n'évite pas les différentes phases de génération de page par script PHP :

- La lecture des scripts PHP.
- L'interprétation des scripts.
- L'exécution des scripts.

Le principe d'optimisation est de limiter le calcul de résultats similaires en stockant les résultats finaux ou intermédiaires les plus gourmands en temps et en traitement.

Ce principe doit permettre de limiter tous les traitements ayant un coût important parmi lesquels :

- De nombreuses interrogations de données.
- Les échanges réseau.
- Le traitement itératif de nombreuses données.
- De nombreux traitements parallèles de mêmes données.
- La lecture et l'écriture de données en parallèle.

1. Mise en place d'un cache de code PHP

L'une des solutions permettant de mettre en œuvre une optimisation rapide est la mise en place d'un cache de code PHP. Il s'agit d'éviter au serveur de passer son temps à interpréter les codes PHP et à les exécuter. Le cache stocke une version pré interprétée des scripts PHP.

L'une des solutions les plus populaires est l'extension PECL APC : Alternative Php Cache.

```
# pecl install apc
```

Cette installation nécessite cependant le redémarrage de votre serveur Apache. Pour des applications de petite ou moyenne taille en terme de nombre de fichiers et d'utilisateurs simultanés, les configurations de base suffiront à obtenir des améliorations plus que mesurables sur votre application.

Le but étant d'augmenter au maximum le taux de réussite du cache pour obtenir l'information, le code suivant est important :

```
<?php
print_r(apc_cache_info());
```

© ENI Editions - All rigths reserved - Algeria Educ

```
?>
```

Il suffit ensuite de diviser la valeur num_hits par la somme de la valeur num_hits ajoutée à la valeur de num_misses.

```
ratio de réussite = num_hits/(num_hits + num_misses)
```

Lorsque votre ratio est supérieur à 90%, votre application gagne largement à utiliser cette extension.

© ENI Editions - All rigths reserved - Algeria Educ

Stratégie d'optimisation des scripts PHP

1. Principes fondamentaux

Le principe qui dirige le choix du positionnement d'un cache est lié directement à l'impact qu'il produira sur les performances en temps d'exécution et de ressources physiques consommées.

Autant il est facile de comprendre qu'une plate-forme qui ne répond plus rapidement aux clients est un problème lié aux performances du code et de l'architecture, autant il est plus difficile de comprendre pourquoi une application qui consomme des ressources l'est tout autant.

Parmi les ressources qui peuvent être consommées, nous trouvons les écritures et les lectures sur disque, la mémoire, le temps de calcul du processeur, de la bande passante réseau et des requêtes vers la base de données.

L'application boulimique en ressources est dangereuse car elle est un frein direct à la notion de montée en charge. Ce genre d'application est marqué par deux types de problèmes majeurs : des temps de traitement élevés et des dégradations de performances dans le temps.

La montée en charge permet à une application d'absorber de plus en plus de clients en offrant le maximum de performances quant au temps de réponse.

Que pensez-vous, par exemple, d'une application en PHP qui a besoin d'un serveur par client ?

De plus, la consommation anormale de ressources est souvent liée à des problématiques de qualité de code PHP. Cette consommation anormale aboutira toujours à une dégradation rapide et massive des temps de réponse avec l'augmentation du nombre de clients.

Rien ne sert de mettre en place un cache sans avoir ciblé vos objectifs au préalable. L'idée de base consiste à déterminer l'origine de la plus grande perte de performance en terme de temps ressource. En effet, la meilleure des pistes consiste à identifier les phases les plus gourmandes en temps ressource. En effet, un script peut répondre en un temps raisonnable. Cependant l'équation suivante est quasiment toujours vraie.

```
Temps de traitements <=> Ressources consommées
```

2. Démarche d'analyse

Réaliser une étude initiale est une stratégie qui se prépare et qui peut suivre un plan tout simple :

Afin de simplifier votre étude initiale, il faut procéder comme suit :

- Identifier tous les traitements.
- Identifier leur fréquence d'appel respective.
- Découper chaque traitement en phases.
- Mettre en place des sondes de mesure de chacune d'elles.
- Collecter les informations.
- Hiérarchiser simplement les priorités.
- Définir les limites de vos actions d'optimisation.
- Généraliser vos optimisations.

a. Identification des traitements

Cette phase est rapide et simple et ne demande pas d'effort surhumain.

© ENI Editions - All rigths reserved - Algeria Educ

Si vous avez une application Web ou un site Web, il s'agit d'identifier la liste de tous les scripts PHP qui peuvent être invoqués par les clients.

Dans cette partie, il ne faut pas ajouter les scripts d'administration qui sont l'œuvre d'une attention particulière et ne doivent pas impacter le fonctionnement et le temps de réponse de vos utilisateurs.

b. Identifier leur fréquence d'appel respective

Cette phase, plus technique que la précédente, consiste à prendre une journée type de l'utilisation de votre application. En effet, le but ici est de calculer le coefficient pondérateur global et de rapporter chaque script à un coefficient général. La plus petite utilisation ayant un coefficient de 1.

La meilleure façon d'y arriver est d'analyser l'ensemble des fichiers d'accès Apache par un script PHP et de générer un fichier texte au format CSV. Le format CSV est un format très pratique, facile à générer, lire et analyser. Il s'agit d'un format ligne/colonne où chaque ligne est séparée par un retour à la ligne et chaque colonne par une virgule ou un point-virgule.

Script d'analyse du fichier d'accès Apache

```
<?php

$fd = fopen("access.txt", "r");

$data=array();
if ($fd) {
    while (!feof($fd)) {
        $buff = fgets($fd, 4096);
    $elt=explode ( " ", $buff);

    if (!isset($elt[6]) or !strstr($elt[6], ".php")) continue;
    if (isset($elt[3])) {
        $date= substr($elt[3],1, 11);
        if (!isset($data[$date]['cpt'])) $data[$date]['cpt']=1;
        else $data[$date]['cpt']++;

        $urls=explode("?", $elt[6]);
        $url=$urls[0];
        if (!isset($data[$date][$url]))
$data[$date][$url]['cpt']=1;
        else $data[$date][$url]['cpt']++;
    }
    }
    fclose($fd);
}

foreach ($data as $d => $f) {
    foreach ($data[$d] as $url => $content) {
        if ($url=='cpt') continue;
        print "$d;$url;".$cpt=$data[$d][$url]['cpt']."\n";

    }
}
?>
```

Fichier résultat de l'analyse

```
27/May/2010;/consultation/index.php;4
27/May/2010;/consultation/navigation.php;4
27/May/2010;/consultation/main.php;2
27/May/2010;/consultation/phpmyadmin.css.php;9
27/May/2010;/consultation/db_structure.php;2
27/May/2010;/consultation/tbl_structure.php;5
27/May/2010;/consultation/tbl_select.php;8
27/May/2010;/superadmin/index.php;2
27/May/2010;/superadmin/navigation.php;1
27/May/2010;/superadmin/import.php;1
27/May/2010;/superadmin/phpmyadmin.css.php;2
```


© ENI Editions - All rigths reserved - Algeria Educ

```
27/May/2010;/userInfo/index.php;5
27/May/2010;/userInfo/user.php;5
28/May/2010;/consultation/index.php;5
28/May/2010;/consultation/navigation.php;4
28/May/2010;/consultation/db_structure.php;2
28/May/2010;/consultation/phpmyadmin.css.php;23
28/May/2010;/consultation/tbl_structure.php;6
28/May/2010;/consultation/tbl_sql.php;1
28/May/2010;/consultation/import.php;4
28/May/2010;/superadmin/tbl_structure.php;6
28/May/2010;/superadmin/phpmyadmin.css.php;15
28/May/2010;/superadmin/navigation.php;1
28/May/2010;/superadmin/tbl_sql.php;3
28/May/2010;/superadmin/import.php;4
28/May/2010;/consultation/querywindow.php;3
28/May/2010;/superadmin/querywindow.php;1
28/May/2010;/consultation/tbl_change.php;1
28/May/2010;/consultation/main.php;2
28/May/2010;/userInfo/index.php;10
28/May/2010;/userInfo/user.php;6
```

c. Découper chaque traitement en phases

Il s'agit d'une phase critique devant fournir le niveau de précision adéquat de chacune des phases.

Chaque script peut être découpé en phases de traitement afin de mesurer l'impact de ces phases dans l'ensemble de l'exécution du script. Pour déterminer ces phases, il faut trouver le juste milieu entre chaque ligne de code et l'ensemble du script. Par exemple, le temps global d'exécution est intéressant et cependant peu pertinent pour mettre le doigt rapidement sur ce qui ne va pas ou ce qui consomme du temps.

Le découpage en phases doit être caractérisé par sa capacité à découper de manière nette les différentes étapes du script et mettre en relief celles-ci par rapport aux étapes similaires dans les scripts voisins.

Les traitements mesurés peuvent être :

- La lecture en base de données
- L'écriture en base de données
- La conversion de modèles (tableau PHP en objet par exemple)
- La mise en forme par le moteur de rendu
- La vérification d'authentification
- …

Le choix des phases doit être homogène entre les scripts afin de faciliter le diagnostic d'anomalie locale. L'anomalie locale est un bug lié à la surconsommation de temps ressource dans un seul script alors que l'ensemble des autres scripts PHP consomme une quantité équivalente ou inférieure de temps ressource. La solution technique est appelée benchmarking et consiste à comparer grossièrement les performances d'un élément par rapport à des éléments similaires.

d. Mettre en place des sondes de mesure pour chaque phase

La stratégie la plus rapide est d'utiliser directement un module existant pour réaliser les mesures. Le besoin est toujours de trouver et de comprendre ce que votre application passe le plus clair de son temps à exécuter.

Pour cela un module très simple permet d'afficher les temps de passage dans différentes parties du script PHP : le module Benchmark du projet PEAR.

Comme pour l'ensemble des modules PEAR, l'installeur se charge d'une tâche un peu complexe : intégrer le module pour une utilisation immédiate dans votre environnement.

Pour installer le module Benchmark, une simple commande suffit :

© ENI Editions - All rigths reserved - Algeria Educ

```
# pear install Benchmark
```

L'utilitaire PEAR se charge de l'ensemble des opérations techniques de récupération, d'installation et d'intégration dans votre installation courante et ceci aussi bien sous environnement Linux que Microsoft Windows.

```
downloading Benchmark-1.2.7.tgz ...
Starting to download Benchmark-1.2.7.tgz (9,506 bytes)
.....done: 9,506 bytes
install ok: channel://pear.php.net/Benchmark-1.2.7
```

e. Exemple d'utilisation du module Benchmark

Voici un script avec une boucle réalisant aléatoirement des attentes qui augmentent avec l'index de la boucle. Cet exemple met en évidence les temps annoncés à l'exécution du script et les résultats obtenus par le module de mesure.

Les résultats peuvent être directement produits au format HTML ou sous forme de tableau associatif PHP.

Le code d'exemple

```
<?
Require_once 'Benchmark/Timer.php';

$timer = new Benchmark_Timer();
$timer->start();
echo "<H1>Execution</H1>";
echo "<lu>";
for ($i=1;$i<11; $i++) {
    $attente=rand(0, $i);
    sleep ($attente);
    echo "<li>Boucle $i: Attente de $attente sec.</li>";
    $timer->setMarker('Boucle_'.$i);
}
echo "</lu>";
$timer->stop();
echo "<hr/>";
echo "<H1>Résultats du module Benchmark::Timer format HTML</H1>";
$timer->display();

echo "<hr/>";
echo "<H1>Résultats du module Benchmark::Timer format PHP</H1>";
echo "<pre>";
print_r($timer->getProfiling());
echo "</pre>";
?>
```

Le résultat du script et des mesures

```
Execution
 Boucle 1: Attente de 1 sec.
 Boucle 2: Attente de 0 sec.
 Boucle 3: Attente de 2 sec.
 Boucle 4: Attente de 1 sec.
 Boucle 5: Attente de 1 sec.
 Boucle 6: Attente de 5 sec.
 Boucle 7: Attente de 3 sec.
 Boucle 8: Attente de 6 sec.
 Boucle 9: Attente de 0 sec.
 Boucle 10: Attente de 5 sec.

Résultats du module Benchmark::Timer format HTML
```

		time index	ex time	%

© ENI Editions - All rigths reserved - Algeria Educ

1	**Start**	1277128289.55000400	-	0.00%
2	**Boucle_1**	1277128290.54945100	0.999447	4.16%
3	**Boucle_2**	1277128290.54951700	0.000066	0.00%
4	**Boucle_3**	1277128292.54956100	2.000044	8.33%
5	**Boucle_4**	1277128293.54962900	1.000068	4.17%
6	**Boucle_5**	1277128294.55003000	1.000401	4.17%
7	**Boucle_6**	1277128299.54997100	4.999941	20.83%
8	**Boucle_7**	1277128302.55012500	3.000154	12.50%
9	**Boucle_8**	1277128308.55048500	6.000360	25.00%
10	**Boucle_9**	1277128308.55053900	0.000054	0.00%
11	**Boucle_10**	1277128313.55084300	5.000304	20.83%
12	**Stop**	1277128313.55088900	0.000046	0.00%
13	**total**	-	24.000885	100.00%

```
Résultats du module Benchmark::Timer format PHP
Array
(
    [0] => Array
        (
            [name] => Start
            [time] => 1277128289.55000400
            [diff] => -
            [total] => -
        )

    [1] => Array
        (
            [name] => Boucle_1
            [time] => 1277128290.54945100
            [diff] => 0.999447
            [total] => 0.999447
        )

   ...

    [10] => Array
        (
            [name] => Boucle_10
            [time] => 1277128313.55084300
            [diff] => 5.000304
            [total] => 24.000839
        )

    [11] => Array
        (
            [name] => Stop
            [time] => 1277128313.55088900
            [diff] => 0.000046
            [total] => 24.000885
        )

)
```

© ENI Editions - All rigths reserved - Algeria Educ

f. Principe de gestion des sondes

Pour faciliter le travail, il est important de trouver le bon moyen de mettre en place les sondes, de manière permanente, sans impacter les performances générales.

Pour cela, il est préférable d'instituer trois principes :

- Toujours réaliser la mise en place de sondes.
- Proposer des labels standardisés.
- Offrir la possibilité de désactiver les sondes.

Le premier principe facilite la mise en place d'indicateurs dès le début. Il est toujours plus difficile d'ajouter quelque chose qui n'existe pas et qui n'a aucun intérêt direct sur les résultats finaux.

Le second principe offre la possibilité d'une visibilité plus importante sur la nature des labels et des briques mesurés. En effet, si chaque développeur identifie des labels différents, cela rend plus complexe une analyse globale des performances.

De même, la normalisation des labels offre une possibilité de réaliser des études de comparaison entre différents scripts afin de détecter les divergences et trouver soit des anomalies à corriger, soit les optimisations spécifiques pouvant être généralisées à l'ensemble des parties de code similaires.

Le troisième principe consiste à offrir un moyen de désactiver les sondes rapidement.

g. Version du code de mesure

Voici donc une version de la gestion des sondes normalisées de l'ensemble des informations sur les labels classiques et des classes de mesure. La création d'instance est relayée vers un fichier générique perfConstant.php qui offre un point d'entrée unique pour les sondes.

```
 <?php

define ("PERF_ACTIVE", 0);
define ("PERF_ATTENTE", "MESURE ATTENTE");
define ("PERF_BOUCLE", "MESURE BOUCLE");
define ("PERF_CONN_BD", "MESURE CONNEXION BD");
define ("PERF_SQL_REQUETE_SQL", "MESURE REQUETE SQL");


if (PERF_ACTIVE) {
    Require_once 'Benchmark/Timer.php';
} else {
    class Benchmark_Timer {
        function start() {}
        function stop() {}
        function setMarker($p=NULL){}
        function display() {}
        function getProfiling() {}
    }
}

$timer = new Benchmark_Timer();
?>
```

Ce code contient une constante PERF_ACTIVE permettant d'activer ou de désactiver la prise de mesure.

Quand la constante PERF_ACTIVE est positionnée à 0, le comportement de la classe est limité et elle est alors transformée en classe vide (c'est-à-dire que les fonctions ne répondent plus à aucun service et ne font plus rien du tout).

Ce fichier contient, de plus, un nombre important de constantes qui doivent être utilisées afin de normaliser les labels au maximum.

Une fois remodelé, le script de mesure précédent devient :

```
<?
```


© ENI Editions - All rigths reserved - Algeria Educ

```
Require_once 'PerfConstant.php';

$timer->start();
echo "<H1>Execution</H1>";
echo "<lu>";
for ($i=1;$i<11; $i++) {
    $attente=rand(0, $i);
    sleep ($attente);
    echo "<li>Boucle $i: Attente de $attente sec.</li>";
    $timer->setMarker(PERF_BOUCLE.'_'.$i);
}
echo "</lu>";
$timer->stop();
if (PERF_ACTIVE) echo "<hr/>";
if (PERF_ACTIVE) echo "<H1>Résultats du module Benchmark::Timer
format HTML</H1>";
$timer->display();

if (PERF_ACTIVE) echo "<hr/>";
if (PERF_ACTIVE) echo "<H1>Résultats du module Benchmark::Timer
format PHP</H1>";
if (PERF_ACTIVE) echo "<pre>";
print_r($timer->getProfiling());
if (PERF_ACTIVE) echo "</pre>";
?>
```

Il est plus générique et rien ne dépend de ce script pour l'activation des mesures.

h. Hiérarchiser les priorités

La stratégie est simple. Il est impératif d'optimiser ce qui est le plus souvent utilisé.

Cela signifie que vous devez travailler prioritairement sur l'amélioration des scripts PHP ayant le plus d'accès.

Il suffit d'effectuer un calcul grossier pour se persuader de ce bien-fondé :

Un script très lent prend 90 secondes en exécution et fonctionne tous les jours une seule fois. Son optimisation de 30 % de temps de traitement va produire un gain de 30 secondes. Il faut aussi noter qu'une fréquence faible d'appel est le signe d'une utilisation faible de la ressource.

Un second script prend 3 secondes en exécution et est appelé 1000 fois par jour par des utilisateurs de votre application. Son optimisation de 10% de temps de traitement, soit 300ms par appel, produira un gain immédiat de 300 secondes.

Le facteur de fréquence est clairement lié avec la notion d'utilisation et d'impact utilisateur. Cela signifie que plus un service est utilisé, plus il est populaire et donc plus il est important qu'il soit optimisé et qu'il réponde avec des temps de réponse faibles et une qualité de réponse élevée.

Dans le chapitre sur le management de la qualité, ce livre présente la méthode ABC ou courbe de Pareto.

i. Définir les limites des actions d'optimisation

Afin d'être les plus performants possible dans cette étape, nous allons utiliser la loi de Pareto décrite dans le chapitre Management de la qualité de la plate-forme sur le management de la qualité.

Cette méthode permet de définir une cible d'optimisation de 80% des résultats globaux.

Pour cet exemple, nous verrons une première analyse consistant à mettre en avant les scripts PHP prioritaires, puis dans une seconde analyse, nous utiliserons la méthode de Pareto pour cibler les parties prioritaires du traitement.

j. Définir les scripts importants à optimiser

Voici donc une première analyse Pareto de la fréquentation des pages et un moyen de déterminer les 80% de scripts prioritaires dans notre étude d'optimisation de traitement.

Dans cette première approche nous déterminons les scripts PHP prioritaires pour notre étude initiale de l'optimisation, c'est-à-dire que nous cherchons les 80% de scripts les plus souvent appelés car nous suivons le principe :

Fréquence d'appel <=> Popularité et impacts utilisateur

© ENI Editions - All rigths reserved - Algeria Educ

À l'aide d'un tableur et des résultats sur les fréquences d'appel des scripts récoltés plus haut dans ce chapitre, nous arrivons rapidement au tableau suivant :

	Date	Script	Fréquence	Somme cumulée	Pourcentage global
1	28/May/2010	/consultation/phpmyadmin.css.php	23	23	23,7%
2	28/May/2010	/superadmin/phpmyadmin.css.php	15	38	39,2%
3	28/May/2010	/userInfo/index.php	10	48	49,5%
4	28/May/2010	/consultation/tbl_structure.php	6	54	55,7%
5	28/May/2010	/superadmin/tbl_structure.php	6	60	61,9%
6	28/May/2010	/userInfo/user.php	6	66	68,0%
7	28/May/2010	/consultation/index.php	5	71	73,2%
8	28/May/2010	/consultation/navigation.php	4	75	77,3%
9	**28/May/2010**	**/consultation/import.php**	**4**	**79**	**81,4%**
10	28/May/2010	/superadmin/import.php	4	83	85,6%
11	28/May/2010	/superadmin/tbl_sql.php	3	86	88,7%
12	28/May/2010	/consultation/querywindow.php	3	89	91,8%
13	28/May/2010	/consultation/db_structure.php	2	91	93,8%
14	28/May/2010	/consultation/main.php	2	93	95,9%
15	28/May/2010	/consultation/tbl_sql.php	1	94	96,9%
16	28/May/2010	/superadmin/navigation.php	1	95	97,9%
17	28/May/2010	/superadmin/querywindow.php	1	96	99,0%
18	28/May/2010	/consultation/tbl_change.php	1	97	100,0%
		Somme	97		

Le tableau contient la liste des scripts pour la journée du 28 mai triée par fréquence décroissante. Dans cette situation le premier script est le script le plus souvent appelé.

La 4e colonne représente la somme cumulée des fréquences d'appel du script et des ses prédécesseurs.

La 5e colonne représente le pourcentage de la somme cumulée par rapport à la somme totale des fréquences.

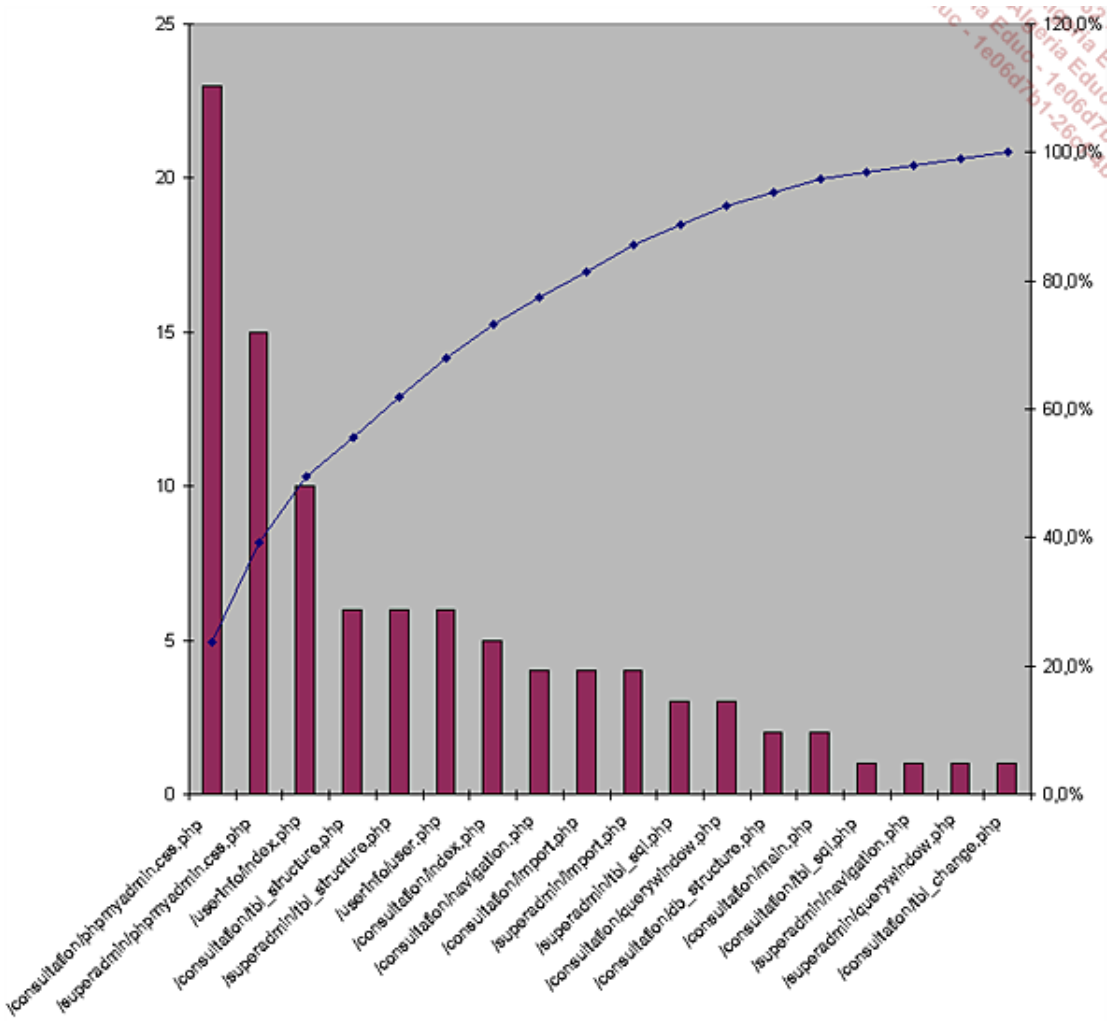
Cela signifie que la fréquence des quatre premiers scripts représente exactement 55,7% des appels.

L'analyse Pareto nous indique que 20% des efforts d'optimisation représentent 80% des gains en performance.

Donc la cible que nous pouvons définir est la limite des scripts représentant 80% de la fréquentation.

Voici donc la représentation sous forme de courbe de Pareto des résultats :

© ENI Editions - All rigths reserved - Algeria Educ



Dans cette optique, nous pouvons focaliser nos actions sur le sous-ensemble de scripts suivant :

	Date	Script	Fréquence	Somme cumulée	Pourcentage global
1	28/May/2010	/consultation/phpmyadmin.css.php	23	23	23,7%
2	28/May/2010	/superadmin/phpmyadmin.css.php	15	38	39,2%
3	28/May/2010	/userInfo/index.php	10	48	49,5%
4	28/May/2010	/consultation/tbl_structure.php	6	54	55,7%
5	28/May/2010	/superadmin/tbl_structure.php	6	60	61,9%
6	28/May/2010	/userInfo/user.php	6	66	68,0%
7	28/May/2010	/consultation/index.php	5	71	73,2%
8	28/May/2010	/consultation/navigation.php	4	75	77,3%
9	**28/May/2010**	**/consultation/import.php**	**4**	**79**	**81,4%**

En focalisant nos efforts d'optimisation sur les neuf premiers scripts PHP, nous allons obtenir 80% des gains de performance. Sachant qu'il y a 18 scripts au total, nous avons donc réduit de 50% notre effort global pour arriver à déterminer ce qui sera le plus efficace.

© ENI Editions - All rigths reserved - Algeria Educ

k. Définir les parties du script importantes à optimiser

Voici donc une seconde analyse Pareto du temps consommé par différentes parties du script PHP et un moyen de déterminer les parties du script produisant 80% de temps consommé. Ces parties et ces traitements seront donc prioritaires dans votre étude d'optimisation du script. Le principe est ici presque une évidence. Réduire la consommation de ce qui est le plus gourmand produit de meilleurs résultats que réduire la consommation de ce qui consomme peu.

En gros, Il est plus facile de généraliser l'utilisation des ampoules basse consommation aujourd'hui que de chercher à diminuer encore la consommation de ces ampoules économiques. Nous ne remettons pas en cause les efforts d'amélioration du produit, seulement la manière d'amener les améliorations à se concrétiser.

À l'aide d'un tableur et des résultats (sur les temps de chaque partie) récoltés plus haut dans ce chapitre, nous arrivons rapidement au tableau suivant :

	Label	Temps	cumul	Pct cumul
1	**Boucle_8**	6,00036	6,00036	25,0%
2	**Boucle_10**	5,000304	11,000664	45,8%
3	**Boucle_6**	4,999941	16,000605	66,7%
4	**Boucle_7**	3,000154	19,000759	79,2%
5	**Boucle_3**	2,000044	21,000803	87,5%
6	**Boucle_5**	1,000401	22,001204	91,7%
7	**Boucle_4**	1,000068	23,001272	95,8%
8	**Boucle_1**	0,999447	24,000719	100,0%
9	**Boucle_2**	0,000066	24,000785	100,0%
10	**Boucle_9**	0,000054	24,000839	100,0%
11	**Stop**	0,000046	24,000885	100,0%
12	**Start**	0	0	0,0%
	total	24,000885		

Le tableau contient la liste des labels des différentes parties du script, triée par temps d'exécution décroissant. Dans cette situation, le premier label correspond directement à la partie du code consommant le plus de temps par rapport à l'ensemble du traitement.

La 3e colonne représente la somme cumulée des temps d'exécution du label et de ses prédécesseurs.

La 4e colonne représente le pourcentage de la somme cumulée par rapport au temps total.

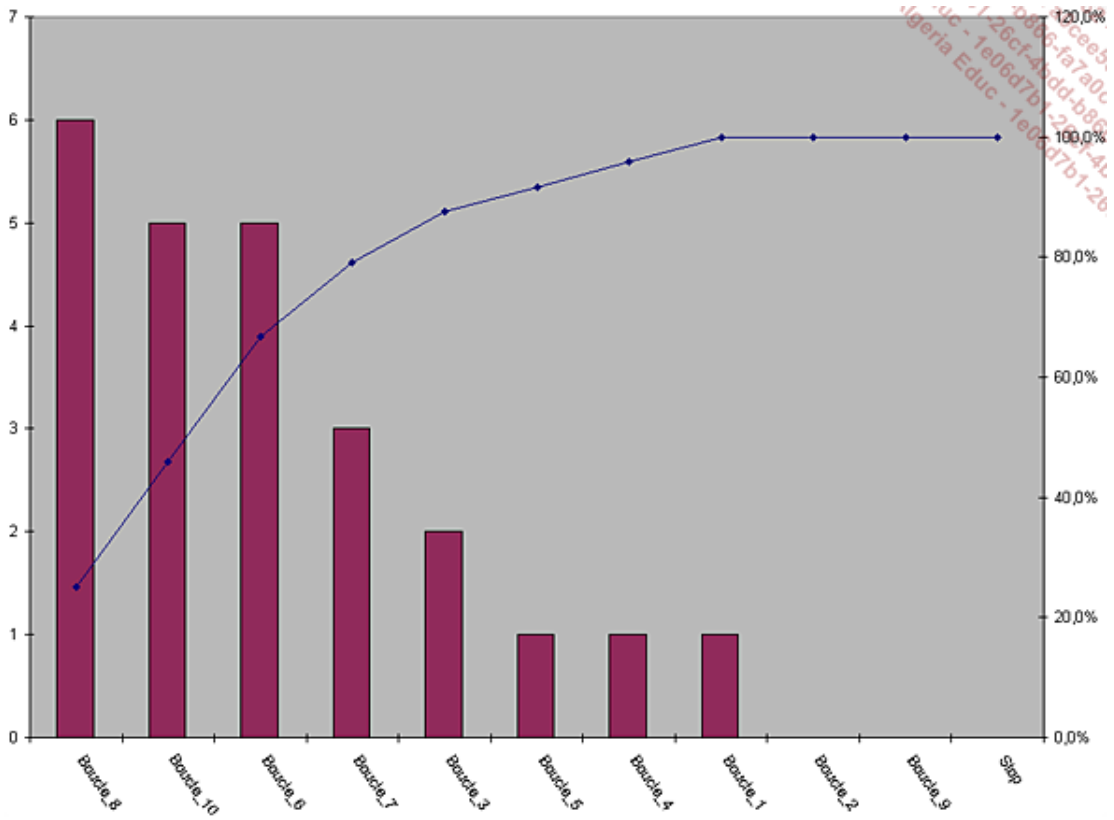
Dans la 2e ligne du tableau, la somme des temps d'exécution cumulés représente exactement 45,8% du temps d'exécution total.

L'analyse Pareto nous indique que 80% du temps d'exécution sont consommés par quatre parties du script.

Donc, la cible que nous pouvons définir est la limite des parties du script représentant 80% du temps d'exécution.

Voici donc la représentation sous forme de courbe de Pareto des résultats :

© ENI Editions - All rigths reserved - Algeria Educ



Dans cette optique, nous pouvons focaliser nos actions sur le sous-ensemble des parties suivantes :

	Label	Temps	cumul	Pct cumul
1	**Boucle_8**	6,00036	6,00036	25,0%
2	**Boucle_10**	5,000304	11,000664	45,8%
3	**Boucle_6**	4,999941	16,000605	66,7%
4	**Boucle_7**	3,000154	19,000759	79,2%

En concentrant nos efforts d'optimisation sur ces quatre parties du script PHP, nous allons obtenir 80% des gains de performance. Sachant qu'il y a 10 parties au total, nous avons donc réduit de 60% notre effort global pour arriver à déterminer ce qui sera le plus efficace.

I. Généralisation des résultats

Nous avons vu qu'une double analyse par la courbe de Pareto permet de définir les actions prioritaires et essentielles à mener.

Il reste cependant à généraliser l'utilisation de cette double analyse avant chaque tentative d'amélioration des traitements. Cette démarche est cependant facilitée par l'existence de scripts de génération des données et de procédés techniques d'obtention de l'information. Il s'agit de ne pas oublier les vertus de la loi de Pareto : se focaliser sur l'essentiel en priorité. Dès que l'essentiel est atteint, vous atteignez un nouveau niveau de qualité pour votre application et vous créez de nouveaux standards pour vos futures réalisations. En appliquant cette démarche, vous ne souhaiterez plus jamais obtenir la qualité de code de votre premier projet en PHP.

Prenez l'habitude de répliquer ce qui marche et a déjà prouvé son efficacité !

La généralisation des optimisations doit être systématique. C'est-à-dire que toute optimisation, si elle offre la possibilité de produire une amélioration générale, doit être modularisée et déployée massivement.

Si vous trouvez une stratégie qui permet de réduire vos temps de traitements de base de données par 10, comme par exemple l'idée de généraliser votre découverte au sein d'une classe gérant à la fois connexion, requêtes et optimisation est une très bonne idée.

© ENI Editions - All rigths reserved - Algeria Educ

Vous aurez alors un autre axe d'amélioration qui se dégagera. Dans une analyse Pareto qui suivra, il se peut que vous trouviez automatiquement la bonne réponse à l'optimisation du script en trouvant un module ou une classe PHP répondant immédiatement à votre besoin.

La démarche de généralisation de vos apports en qualité développe automatiquement le potentiel de qualité latent de votre application. Ce potentiel latent sera la base permettant de produire plus rapidement, moins cher et de meilleure qualité des logiciels en PHP. Les ressources libérées produiront une force pour améliorer encore la qualité. Il s'agit du cercle vertueux de la qualité !

© ENI Editions - All rigths reserved - Algeria Educ

Introduction à la qualité dans le code PHP

La qualité est souvent vue comme une démarche lourde, contraignante et incertaine. En effet, qu'est-ce que la qualité dans le code PHP ? Est-ce sa performance ? Est-ce son évolutivité ? Sa documentation ? Son respect des normes de codage ?

La qualité n'est rien d'autre que ce que l'on veut bien qu'elle soit ! L'art du bon gestionnaire développeur, manageur ou intégrateur est de définir clairement la notion de qualité et de poser des indicateurs (de préférence automatiques, peu intrusifs et peu coûteux à mettre en place). Puis de simplement suivre les indicateurs, de corriger les décalages de qualité au fur et à mesure que ceux-ci se présentent et d'y apporter des réponses adéquates.

La qualité s'appuie donc sur cinq piliers :

- La définition claire des points de qualité.
- La mise en place d'indicateurs pertinents.
- Le suivi des indicateurs.
- La prise de décision suite aux décalages.
- Le maintien du processus de qualité dans le temps.

Cette méthode n'est qu'une déclinaison de la roue de Deming expliqué dans le chapitre Management de la qualité de la plate-forme.

Nous allons donc présenter dans ce chapitre des axes de qualité classiques pour un projet de développement en PHP afin de mener, dès le démarrage du projet, les actions qui feront la différence « qualité » de votre résultat final.

Parmi tous les axes de qualité, nous pouvons citer :

- La cohérence d'écriture du code.
- Les commentaires.
- La complexité du code.
- La pertinence des tests unitaires.
- La pertinence des tests fonctionnels.
- La couverture des tests unitaires.
- La couverture des tests fonctionnels.
- La documentation des composants.
- L'évolutivité de l'architecture logicielle.
- Les temps de réponse.
- La capacité de montée en charge.
- La stabilité en fonctionnement.
- La polyvalence des équipes de développement.
- La centralisation de l'information.

© ENI Editions - All rigths reserved - Algeria Educ

- La robustesse de l’application PHP.
- La pertinence des réponses.
- La robustesse du code face aux erreurs.
- La pertinence des messages d’erreur.

Ceci n’est pas une liste exhaustive de tous les axes possibles. En effet, dans certains domaines très particuliers comme la défense, l’énergie ou les logiciels médicaux, de nouveaux axes de qualité sont nécessaires. Par exemple, il n’est pas possible que le logiciel fonctionne dans 99,99% des cas dans le choix d’un mélange de composants chimiques pour fabriquer un médicament destiné à des millions d’individus. Dans ce cas, la qualité attendue est de 100% et donc un nouvel axe de qualité doit être positionné permettant de valider systématiquement le bon résultat : cet indicateur peut donc s’intituler « Prédictibilité totale du résultat ».

Ce mécanisme de choix ou de création d’indicateur est de votre responsabilité autant que l’application des autres étapes.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction à la gestion de la qualité des traces

Parmi tous les axes de qualité, nous pouvons en citer quelques-uns qui sont directement liés à la qualité des traces :

- La cohérence d'écriture du code.
- Les commentaires.
- La complexité du code.
- La couverture des tests unitaires.
- La documentation des composants.
- L'évolutivité de l'architecture logicielle.
- La stabilité en fonctionnement.
- La polyvalence des équipes de développement.
- La centralisation de l'information.
- La robustesse de l'application PHP.
- La robustesse du code face aux erreurs.
- La pertinence des messages d'erreur.

Pour obtenir un système capable d'offrir une base solide pour le dépistage et l'analyse des erreurs, il est impératif que votre système produise des traces applicatives ayant des propriétés définies :

1. Normalisation du format

L'ensemble des traces doit posséder une forme, ou mieux un format unique permettant aux équipes d'analyse de déterminer rapidement la nature, le contexte et le niveau de priorité d'erreur.

Il est plus facile à la fois pour des humains et pour des programmes d'analyse de déduire l'information essentielle du message si celui-ci possède une forme classique. Par analogie, il est plus facile de parler toujours la même langue pour vos diverses activités de la journée (travail, courses, famille, ami, etc.) plutôt que de parler une langue différente à chacune de vos activités.

Il en est de même pour vos applications. Plus les traces sont similaires dans le format et plus il est facile d'être pertinent, efficace, performant et rapide dans l'analyse et dans les réponses à donner à chacun des messages.

2. Exhaustivité des traces

Il est important de ne jamais masquer une erreur dans votre code applicatif. Il est préférable de corriger en priorité l'origine des problèmes générant le plus d'erreurs et, dans un contexte plus contraignant, de réaliser un filtrage des fichiers d'erreurs afin de masquer au niveau des consoles d'administration les erreurs sans les supprimer du système.

En effet, un message d'erreur masqué représente une faute de code, de conception ou d'architecture de déploiement. Dans cette optique, il est important de maîtriser l'origine de la production de l'erreur et d'y apporter une réponse adaptée (comme une correction du code, un paramétrage applicatif ou un paramétrage des serveurs).

3. Complétude des traces

© ENI Editions - All rigths reserved - Algeria Educ

Pour être exploitable, il est important qu'une trace soit la plus complète possible. Cela signifie qu'elle contient l'ensemble des informations permettant de localiser le code qui a généré l'erreur rapidement, ainsi que le contexte ou l'enchaînement d'actions ayant amené le système à produire l'erreur.

Dans ce contexte, il est primordial, si ce n'est impératif, de greffer aux différents messages d'erreur l'ensemble des informations concernant le contexte. Le contexte d'une erreur consiste à associer :

- Les informations sur l'utilisateur.
- Les informations sur la fonctionnalité utilisée.
- Les données saisies ayant produit l'erreur.
- La date de l'événement.

Il sera donc ensuite possible de recouper les erreurs liées à des indisponibilités de service technique et de programmer, si nécessaire, un mode de repli permettant de limiter l'impact sur l'expérience utilisateur.

La correction devra donc prendre en compte l'erreur et trouver une stratégie permettant de répondre au mieux à la demande ou afficher un message permettant à l'utilisateur de comprendre l'origine du problème.

Dans le cas où l'erreur n'est pas d'origine technique mais plutôt due à l'association systématique d'une demande particulière avec des données spécifiques, alors la correction peut se traduire soit par une modification de code, soit par une correction, modification, ajout ou suppression d'informations dans la base de données afin de produire un résultat correct lors des futures utilisations de l'application dans ce même contexte.

L'histoire nous rappelle que des logiciels comme Microsoft Word dans ses versions initiales venaient à s'arrêter à l'ouverture de certains documents Word. En effet, l'association d'une application avec certaines données données produit des dysfonctionnements parfois sévères tels que des ralentissements, des surconsommations de ressources ou des arrêts d'application. Ce type de dysfonctionnement apparaît généralement lors de traitements de grands volumes de données ou dans un système fortement sollicité.

4. Concentration des traces

Les architectures et les environnements techniques devenant de plus en plus complexes, il est important de fournir la possibilité de croiser les informations et les traces provenant de plusieurs systèmes.

Pour ce faire, il est possible de consommer les fichiers au fur et à mesure. Cependant, il existe une stratégie très simple et éprouvée basée sur les serveurs de gestion des traces existant pour la plupart dans les systèmes d'exploitation.

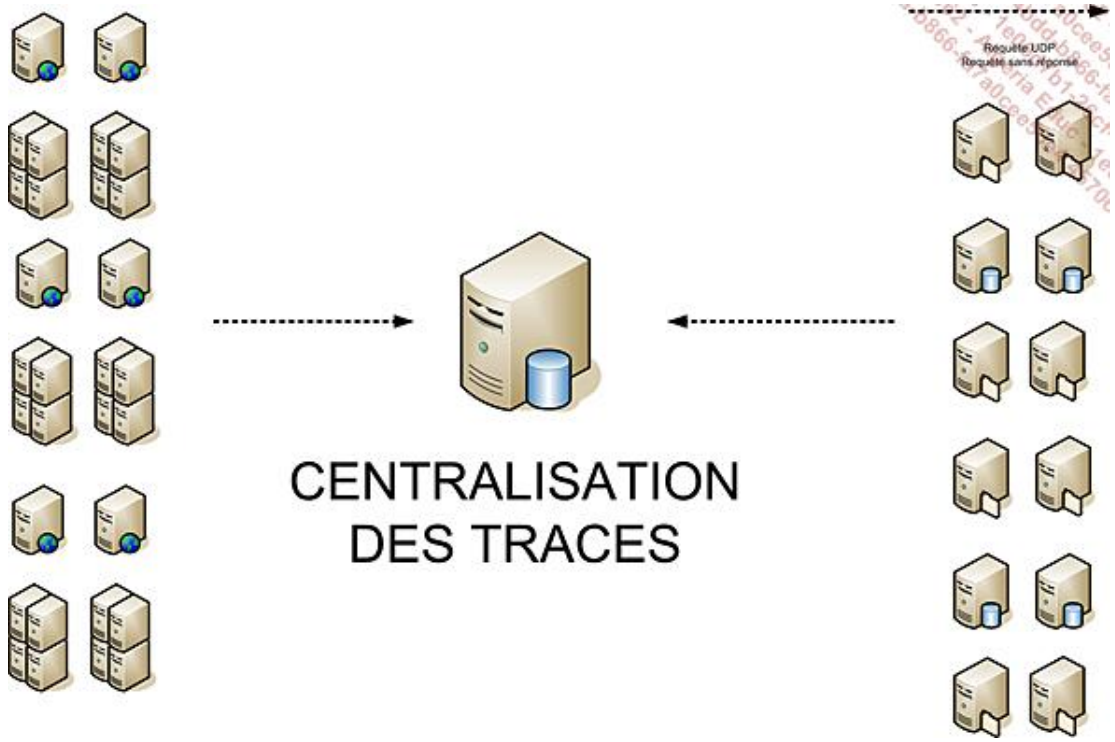
Voici un exemple de paramétrage pour PHP et le démon Syslog permettant de renvoyer les informations vers un serveur centralisé tout en conservant un fichier de traces localement pour analyse.

L'avantage le plus évident de l'utilisation d'un serveur Syslog est qu'il s'appuie sur une technologie en mode non connecté : UDP. Cela signifie que si le serveur Syslog ou le serveur centralisé ne répondent plus, alors le service continue à fonctionner car les traces sont envoyées sans attente de validation de bonne réception des messages. Il s'agit d'un mode non fiable qui offre la possibilité de garantir un haut niveau de performance et de sécurité tout en activant l'idée magique d'un point unique de contrôle de l'ensemble de vos applications PHP.

La centralisation des messages en utilisant le protocole UDP offre la sécurité concernant l'indisponibilité du serveur de traces. L'envoi des traces n'est jamais acquitté et ne produira donc jamais un ralentissement de votre service sur le Web

---

© ENI Editions - All rigths reserved - Algeria Educ



Code PHP pour l'envoi des traces en local

```
<?php
$nomApplication= «MonApplication » ;
$dateFormat= « Y/m/d_H:i:s » ;
$Tuyau= LOG_LOCAL5 ;
$syslogOptions=LOG_ODELAY;
function log($priority, $msg) {

    openlog($nomApplication, $syslogOptions, $tuyau);

    $dtxt=date($dateFormat) ;
    foreach (split("\n", $msg) as $lmsg) {
      $message= "$dtxt;$lmsg" ;
      syslog($priority ,$message);
    }
    closelog();
}
?>
```

Code PHP pour l'utilisation locale

```
< ?php
require_once ('trace.lib.php');

log( LOG_INFO, "DEBUT du script");
echo « fin du traitement. » ;
log( LOG_INFO, "FIN du script");

?>
```

Configuration locale du serveur Syslog dans le fichier /etc/syslog.conf

```
none.local5        /var/log/messages
*.local5           /var/log/scriptPhp.log
*.local5           @serveur.central.local
```

Ici la configuration indique qu'il ne faut pas écrire les traces du tuyau local5 dans le fichier de messages

© ENI Editions - All rigths reserved - Algeria Educ

général /var/log/messages. Ces traces seront écrites dans le fichier /var/log/scriptPHP.log. La dernière directive avec @ indique que les traces du tuyau local5 seront envoyées par UDP (mode déconnecté) vers le serveur dont le nom réseau est « serveur.central.local ».

Configuration du serveur Syslog central

Le serveur central « serveur.central.local » devra garantir le lancement du service Syslog avec l'option permettant la réception des messages à distance.

Le fichier de configuration /etc/sysconfig/syslog doit contenir :

```
SYSLOGD_OPTIONS= « -r -m 0 »
```

L'option -m 0 désactive le marquage des messages. L'option -r active la réception à distance.

Si vous ne trouvez pas le fichier /etc/sysconfig/syslog, consultez la documentation de votre distribution Linux afin de trouver l'endroit explicite pour activer ces deux paramètres.

Ces paramètres se retrouvent dans la plupart des distributions à base de paquets RPM telles que Red Hat, Fedora Linux, Mandrake Linux, etc.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction aux tests en charge

Parmi tous les axes de qualité dans les tests en charge, nous pouvons en citer quelques-uns :

- La complexité du code.
- L'évolutivité de l'architecture logicielle.
- Les temps de réponse.
- La capacité de montée en charge.
- La stabilité en fonctionnement.
- La pertinence des réponses.

Le test en charge a deux buts très simples :

- Connaître le niveau de performance des réponses du système.
- Connaître la capacité à répondre durablement dans le temps.

Les tests en charge ont pour but de dessiner et de quantifier le niveau de performance générale d'une application PHP.

1. Niveau de performance

Il s'agit de la garantie que les réponses sont correctes et que le délai de fourniture des réponses est raisonnable.

Tout système a une limite, à vous de la connaître !

La limite de votre système se nomme point de rupture. Il s'agit d'un ensemble de valeurs d'indicateurs définissant le niveau de performance maximum. Dès que ce niveau est dépassé, la consommation des ressources va augmenter très rapidement et le niveau de performance va diminuer significativement.

a. JMeter, l'outil libre de test en charge

L'outil le plus simple et le plus fonctionnel actuellement disponible dans l'offre du logiciel libre est JMeter. JMeter est un injecteur de trafic écrit en Java.

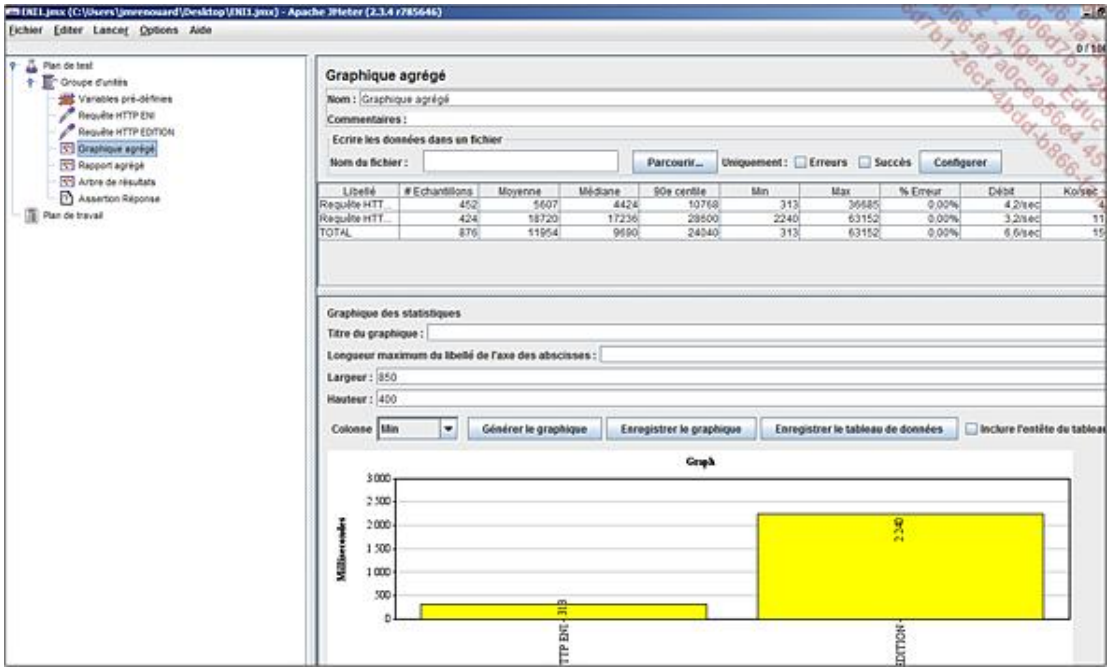
Il est disponible sur le site du projet Apache Jakarta.

http://jakarta.apache.org/jmeter

Dans l'exemple qui suit, nous avons défini un groupe de 100 utilisateurs effectuant chacun 100 requêtes sur http://www.eni.fr et 100 requêtes sur htttp://www.editions-eni.fr/Formations/

Nous avons ajouté une recherche de la chaîne « Editions ENI » dans les deux pages et avons collecté les résultats sous forme de rapport agrégé avec graphique.

© ENI Editions - All rigths reserved - Algeria Educ



Interface JMeter : Graphique agrégé des temps minimum

b. Les interfaces des connecteurs d'injection

JMeter est un injecteur ayant différents connecteurs permettant de simuler du trafic sur plusieurs types de connexion tels que :

- HTTP.
- HTTPS.
- FTP.
- Base de données.
- LDAP.

Il existe, de plus, une solide base de code permettant d'écrire vos propres connecteurs.

Il permet aussi de simuler plusieurs connexions parallèles avec des temps de lancement et de montée en régime nominal d'injection.

c. Les interfaces de mesure

Il existe aussi un nombre important de moyens de contrôler les résultats :

- Graphique, avec temps de réponse maximum et minimum.
- Tableau des réponses.
- Tableau des réponses en erreur.
- Rapport agrégé des valeurs.
- Sortie de résultat en XML, CSV ou autres formats texte.

© ENI Editions - All rigths reserved - Algeria Educ

d. Les interfaces de définition des tests

Les tests en charge peuvent être complexes avec des phases d'authentification, de consultation, de modification et de navigation dans un large volume de données.

Il est donc important que l'outil de test en charge permette de définir ses scénarios de tests en charge de manière efficace et permette aux équipes de développement de valider, avant le lancement d'une nouvelle version, l'adéquation des performances avec l'utilisation possible et attendue de l'application.

Il existe donc de nombreux éléments permettant de réaliser toutes les opérations classiques d'un langage de programmation basique tels que :

- La définition de variables locales et globales.
- Les conditions si…alors..sinon.
- Les boucles « Pour Chaque » et « Tant Que ».
- Les conditions aléatoires.
- Les assertions de validité de résultats.

e. Détection du point de rupture avec JMeter

Le point de rupture est une valeur qui ne concerne que le nombre d'utilisateurs en parallèle de votre test en charge. Il est à noter cependant qu'un paramètre « Durée de montée en charge » est présent dans le groupe d'unités afin de ne pas créer un pic initial trop important dans votre système.

Le point de rupture est le nombre d'utilisateurs effectuant une série simple ou composée d'opérations sur votre application avant que l'application ne puisse plus fournir de réponse exacte dans un temps acceptable.

Ce point de rupture est en rapport avec l'environnement de test en charge de votre application et est aussi relatif au scénario simple ou composé que chaque utilisateur simulé va effectuer sur votre application.

Le but du test en charge est de définir le nombre maximum d'utilisateurs possible sur le système et sur l'application ainsi que le nombre de requêtes par seconde que cet ensemble serveur Web + application Web est capable de supporter.

Le premier critère permet de définir le nombre de serveurs idéalement nécessaires pour votre application et le second sert d'indicateur de performance général pour vos futures versions. En effet, la gestion de la qualité passe par la possibilité d'offrir plus de services tout en gardant une capacité de débit constant entre les versions.

Il est facilement envisageable de perdre un peu de débit à chaque version jusqu'à un point de non-retour où trop d'optimisation et de corrections seraient alors nécessaires pour améliorer le service.

2. Durabilité du niveau de performance

Il se peut que dans le temps les performances se dégradent doucement ou que la consommation de ressources augmente peu à peu.

Dans la plupart des tests en charge, le test de vieillissement est souvent oublié.

Le test de vieillissement consiste à injecter 80% du niveau maximum de performance ou point de rupture. Dès que ce point est défini, on réalise un test de plusieurs heures (entre 6 et 24 heures) permettant de mettre en lumière la dérive éventuelle de consommation.

Il n'est pas rare de mettre en place une application qui « tient » les temps de réponse mais qui va peu à peu dériver pour devenir médiocre. Les serveurs Apache doivent être redémarrés régulièrement parce que l'application "vieillit" mal et la qualité de service se dégrade dans le temps.

Le test de vieillissement est important car il est un indicateur fort de qualité de votre code. En effet, voici une liste des origines possibles des causes de mauvais vieillissement :

- Gestion de l'ouverture de connexion réseau.
- Choix des moyens de stockage.

© ENI Editions - All rigths reserved - Algeria Educ

- Gestion de l'ouverture des ressources.
- Algorithmique des traitements inadapté aux volumes des données.
- Fuite mémoire dans la gestion de mémoire.
- Synchronisation de divers traitements.

Dans ce type de test de longue durée, il est impératif de suivre les indicateurs suivants :

- Consommation mémoire d'Apache.
- Consommation mémoire du système.
- Consommation d'espace disque.
- Vérification des échanges réseau.
- Nombre de processus en attente sur le système (charge serveur).

Pour effectuer ce type de test, il suffit de prendre la valeur du nombre d'utilisateurs de votre point de rupture en test en charge pur et de positionner cette valeur à 80% de la valeur du point de rupture. Puis d'augmenter de manière significative la valeur du nombre de requêtes par utilisateur. Cette augmentation engendrera une exécution de l'ensemble plus longue dans le temps et permettra de réaliser le test de vieillissement qui offrira de la visibilité sur l'évolution du comportement de l'application dans le temps.

Ce test est primordial, au même titre que le test en charge, car une application « qui vieillit mal », c'est-à-dire qui voit ses performances se dégrader dans le temps, est une application qu'il faudra redémarrer régulièrement et qui offrira à ses utilisateurs des résultats très lents.

Le test de vieillissement, quoique négligé, est aussi important que le test en charge !

© ENI Editions - All rigths reserved - Algeria Educ

Introduction à l'analyse de code

Parmi tous les axes de qualité impactés, nous pouvons citer :

- La cohérence d'écriture du code.
- Les commentaires.
- La complexité du code.
- La couverture des tests unitaires.
- La couverture des tests fonctionnels.
- La documentation des composants.
- L'évolutivité de l'architecture logicielle.
- Les temps de réponse.
- La capacité de montée en charge.
- La stabilité en fonctionnement.
- La polyvalence des équipes de développement.
- La centralisation de l'information.
- La robustesse de l'application PHP.
- La pertinence des réponses.
- La robustesse du code face aux erreurs.
- La pertinence des messages d'erreur.

Une analyse de code avec chacun des trois outils vous permet de mieux comprendre comment améliorer votre code, votre manière de coder ou de développer une application dans son ensemble et vous permet à titre individuel de renforcer les meilleures aptitudes dans votre métier.

Le travail d'analyse et de compréhension des aspects statiques de votre réalisation technique vous permet d'atteindre un niveau supérieur de compétences permettant de détecter des aspects de cohérence et de complexité dans vos réalisations.

1. L'outil d'analyse orienté sécurité : Rats

Rats est un acronyme signifiant outil d'audit brut pour la sécurité.

Rats est un logiciel libre sous licence libre GNU GPL écrit en langage C.

Les versions binaires sur le site ne sont disponibles que pour Windows. Cependant, une compilation sous Linux avec un compilateur GCC ne doit pas poser de problème a priori.

Le but est de mettre en évidence les failles potentielles d'une application PHP et de les indiquer clairement.

À noter que la version courante de Rats est la version 2.3, disponible à l'adresse suivante :

http://www.fortify.com/security-resources/rats.jsp

La base des vulnérabilités est au format XML et est, pour une fois, facile à lire. Voici par exemple, la déclaration d'une

© ENI Editions - All rigths reserved - Algeria Educ

vulnérabilité très connue sur la fonction mail dans le fichier rats-php.xml fourni avec le Rats.

```
<Vulnerability>
    <Name>mail</Name>
    <Info>
      <Severity>High</Severity>
      <Description>
        Arguments 1, 2, 4 and 5 of this function may be passed to
an external
        program. (Usually sendmail). Under Windows, they will be
passed to a
        remote email server. If these values are derived from user
input, make
        sure they are properly formatted and contain no unexpected
characters or
        extra data.
      </Description>
    </Info>
  </Vulnerability>
```

Les deux étapes de fonctionnement de RATS :

- Analyse de l'ensemble des fichiers sources ;
- Affichage du rapport par ordre de risque (Élevé, Moyen, Bas).

Pour cet exercice, j'ai utilisé le code source d'un CMS fonctionnant avec Smarty, le moteur de Template décrit dans ce livre.

J'ai donc décompressé l'ensemble du projet dans le répertoire courant et j'ai lancé la commande suivante.

```
c:\rats-2.3\rats.exe -w 3 . > result.txt
```

L'option -w 3 active le niveau maximum de log. La commande redirige l'intégralité du résultat dans le fichier result.txt vous permettant de conserver le résultat après analyse.

Rats explore plus de 55 vulnérabilités connues dans le langage PHP.

L'analyse de la version 1.7.7 de CMSMS est disponible depuis la page de téléchargement suivante :

http://dev.cmsmadesimple.org/project/files/6#package-1

L'analyse statique effectuée par Rats est très riche en informations :

Les informations disponibles dans le rapport sont les suivantes :

1. Affichage du chargement des bases de vulnérabilités.

```
Entries in perl database: 33
Entries in ruby database: 46
Entries in python database: 62
Entries in c database: 334
Entries in php database: 55
```

2. Liste des fichiers analysés.

```
...
Analyzing ./admin/multicontent.php
Analyzing ./admin/multistylesheet.php
Analyzing ./admin/multitemplate.php
...
```

3. Les informations sur les positions dans les fichiers des fonctions vulnérables.

```
...
./lib/misc.functions.php:1489: Low: is_writable
A potential TOCTOU (Time Of Check, Time Of Use) vulnerability
exists.  This is
the first line where a check has occured.  No matching uses were
detected.
...
```


© ENI Editions - All rigths reserved - Algeria Educ

```
./lib/config.functions.php:47: Low: dirname
./lib/config.functions.php:47: Low: dirname
./lib/config.functions.php:367: Low: dirname
...
```

4. Statistiques des analyses

```
Total lines analyzed: 264369
Total time 0.750000 seconds
352492 lines per second
```

Il est donc important de comprendre que Rats cherche avant tout des appels à des fonctions sensibles dans votre code et ne fait aucune hypothèse sur le contexte d'appel ni sur la manière dont vous appelez ces fonctions. Rats recherche des fonctions vulnérables et ne prend pas en compte les modifications visant à sécuriser ces appels vulnérables dans votre code. Il s'agit ici d'une limite forte de l'outil d'analyse.

Cependant, Rats a le grand mérite de mettre le doigt sur les parties sensibles de votre code.

Le principe à retenir, s'il n'y en a qu'un, est de concentrer dans un point unique de votre application les appels sensibles afin de sécuriser une fois pour toutes les appels et effectuer un maximum de tests et contrôles pour garantir une utilisation adéquate des fonctions vulnérables.

Contrôlez et centralisez en un point unique l'appel de vos fonctions sensibles !

2. L'outil d'analyse des points de vulnérabilités : Pixy

Pixy est un programme permettant de détecter finement les vulnérabilités aux attaques de type XSS et SQLI. Pour rappel, les attaques XSS permettent de modifier les réponses fournies au client pour offrir des accès et des droits. Les attaques SQLI sont des attaques visant les bases de données et l'injection de code SQL au travers de l'interface Web.

L'utilitaire Pixy permet donc de détecter en dehors de toute conception de code les erreurs de programmation offrant des failles de sécurité lors de l'utilisation du code.

Pixy se trouve à l'adresse suivante :

http://pixybox.seclab.tuwien.ac.at/pixy/index.php

3. L'outil d'analyse de qualité d'écriture du code : PHPLint

PHPLint est un analyseur de code ayant son propre parseur et supportant la version 5.3 de PHP.

L'analyseur PHPLint offre les fonctionnalités suivantes :

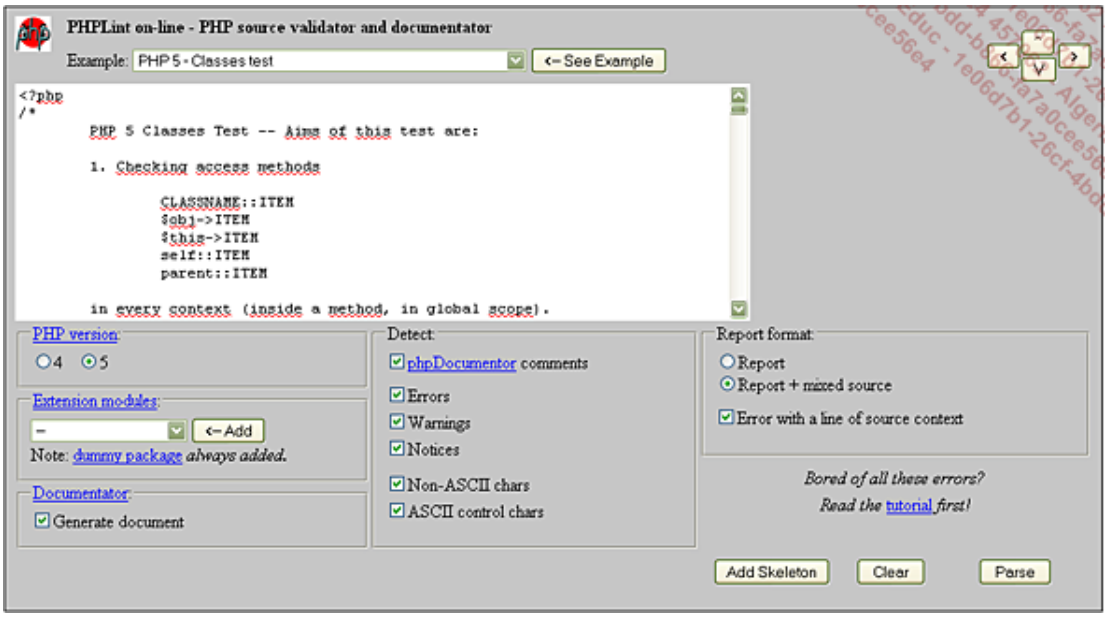
- Validation du typage fort dans PHP.
- Liste des extensions nécessaires pour votre application.
- Détection des erreurs de syntaxe.
- Détection du code mort (code jamais appelé).
- Support de tag de phpDocumentor.
- Détection des variables non utilisées.
- Détection des failles de sécurité.
- Génération de documentation à partir des sources.

L'application PHPLint se trouve à l'adresse suivante : http://www.icosaedro.it/phplint/

© ENI Editions - All rigths reserved - Algeria Educ

PHPLint constitue pour vous un véritable outil de qualité pour détecter les problèmes de code et aussi un outil de suivi de l'évolution de la qualité du code durant le cycle de réalisation.

PHPLint offre, de plus, un service en ligne permettant son utilisation directement dans votre navigateur. Il s'agit d'une alternative très intéressante afin de valider la qualité du code d'une de vos classes le plus rapidement possible.



© ENI Editions - All rigths reserved - Algeria Educ

Introduction aux tests unitaires

Les tests unitaires sont de simples et très courts programmes permettant de mettre en lumière le fonctionnement d'une partie de l'ensemble d'une application ou d'un composant logiciel.

Parmi tous les axes de qualité, nous pouvons citer :

- La pertinence des tests unitaires.
- La couverture des tests unitaires.
- La documentation des composants.
- L'évolutivité de l'architecture logicielle.
- La centralisation de l'information.

Il est possible d'approcher l'ensemble des réalisations techniques par une approche qualité et des tests de contrôle. Cette approche est nommée « Approche pilotée par les tests ». Son principe repose sur le postulat qu'il est toujours plus simple de maintenir et de valider un composant logiciel ayant une interface ou un contrat d'utilisation simple.

Pour avoir le contrat d'utilisation le plus pratique possible, il suffit d'écrire un court programme utilisant l'interface du composant et validant ses principales fonctionnalités.

Afin de résumer cette approche, commencez toujours par écrire vos tests (qui sont ni plus ni moins des programmes de validation de votre composant) puis écrivez votre composant en lui-même.

Cette démarche va à contre-courant des idées reçues sur la réalisation de code. En effet, l'idée commune nous incite à écrire le code du composant en premier puis les tests unitaires, ce qui pose de nombreux problèmes.

Un composant peut être mal écrit la première fois pour des raisons multiples : inexpérience des développeurs, pression sur les délais de livraison, pression sur le volume des fonctionnalités attendues.

Dans ce cadre, les contrats d'utilisation de vos composants (c'est-à-dire la manière de les utiliser dans un autre programme) sont souvent définis au fil de l'eau.

Un composant logiciel mal désigné est toujours difficile à tester ! Écrire le code d'utilisation de votre composant va vous permettre de prendre le recul suffisant afin de vous permettre de réfléchir à l'utilisation de votre composant et de définir le contrat d'utilisation optimal dans votre contexte.

Afin d'achever cette introduction, rappelez-vous toujours qu'un composant ayant une interface ou un contrat d'utilisation simple et intuitif est toujours moins coûteux à maintenir. Sachant aujourd'hui que les coûts de maintenance des logiciels sont les premiers postes de dépenses dans le cycle de vie d'un logiciel, il semble important de partir sur l'hypothèse que, plus ces logiciels sont bien conçus, plus la maintenance sera facilitée et rapide.

1. Tests unitaires

Un test unitaire est un programme très court souvent intégré dans un framework ou un module de test. Ce module de test permet de préparer un environnement de test complet et de lancer automatiquement un ensemble de tests cohérents et indépendants.

2. Cas de PHPUnit3

a. Présentation PHPUnit 3

PHPUnit3 est un framework permettant de réaliser des classes de test de vos applications. Ces tests ont pour objectif de permettre de lancer des tests unitaires en masse et aussi de produire des statistiques sur le passage et la non-régression du fonctionnement du code.

b. Installation de PHPUnit 3

```
pear channel-discover pear.phpunit.de
```

© ENI Editions - All rigths reserved - Algeria Educ

```
pear channel-discover pear.symfony-project.com
pear install phpunit/PHPUnit
```

c. Exemple de code

Soit une classe Utilisateur ayant un attribut nom.

Code de la classe Utilisateur

```
<?php

class Utilisateur
{
    private $nom;

 public function __contruct($nom) {
    $this->nom=$nom;
}
public function getNom() { return $this->nom;}
}
?>
```

Code de la classe PHPUnit de test

```
<?php

require_once 'PHPUnit/Framework.php';
 require_once 'Utilisateur.class.php';

class UtilisateurTest extends PHPUnit_Framework_TestCase
{
    public function testCreation()
    {
        $util=new Utilisateur("ENI");
        $this->assertNotNull($util);
        $this->assertType('object', $util) ;
        $this->assertObjectHasAttribute('nom', $util);

        $this->assertEquals("ENI",$util->getNom());
     }
}
?>
```

Résultat du lancement de phpUnit

```
# phpunit UtilisateurTest
PHPUnit 3.4.4 by Sebastian Bergmann.

F

Time: 0 seconds

There was 0 failureFAILURES!
Tests: 1, Assertions: 4 Failures: 0
```

d. Ensemble des fonctions de test PHPUnit

Fonctions de test	Fonctions inverses	Description
assertArrayHasKey()	assertNotArrayHasKey()	Test de la présence de la clé dans un tableau PHP.

© ENI Editions - All rigths reserved - Algeria Educ

assertClassHasAttribute()	assertNotClassHasAttribute()	Test de la présence de l'attribut dans une classe PHP.
assertClassHasStaticAttribute ()	assertNotClassHasStaticAttribute ()	Test de la présence de l'attribut statique dans une classe PHP.
assertContains()	assertNotContains()	Test de la présence de l'élément dans un ensemble PHP.
assertContainsOnly()	assertNotContainsOnly()	Test de la présence de l'élément uniquement dans un ensemble PHP.
assertEqualXMLStructure()	assertNotEqualXMLStructure()	Test d'égalité de chaîne XML.
assertEquals()	assertNotEquals()	Test d'égalité de contenu.
assertFalse()	assertNotFalse()	Test que la valeur passée en argument est égale à faux.
assertFileEquals()	assertNotFileEquals()	Test que le contenu de deux fichiers est équivalent.
assertFileExists()	assertNotFileExists()	Test d'existence de fichier.
assertGreaterThan()	assertNotGreaterThan()	Test de comparaison numérique plus grand que.
assertGreaterThanOrEqual()	assertNotGreaterThanOrEqual()	Test de comparaison numérique plus grand que ou égal.
assertLessThan()	assertNotLessThan()	Test de comparaison numérique plus petit que.
assertLessThanOrEqual()	assertNotLessThanOrEqual()	Test de comparaison numérique plus petit que ou égal.
assertNull()	assertNotNull()	Test de non nullité.
assertObjectHasAttribute()	assertNotObjectHasAttribute()	Test de la présence d'un attribut par un objet PHP.
assertRegExp()	assertNotRegExp()	Test de la validité d'une expression régulière.
assertSame()	assertNotSame()	Test d'égalité en contenu et en type.
assertSelectCount()	assertNotSelectCount()	Test d'égalité de dénombrement d'élément.
assertSelectEquals()	assertNotSelectEquals()	Test d'égalité de contenu d'élément.
assertSelectRegExp()	assertNotSelectRegExp()	Test d'égalité de contenu d'élément sélectionné par expression régulière.
assertStringEndsWith()	assertNotStringEndsWith()	Test d'égalité de dénombrement d'élément.
assertStringEqualsFile()	assertNotStringEqualsFile()	Test d'égalité de fin de chaîne.
assertStringStartsWith()	assertNotStringStartsWith()	Test d'égalité de début de chaîne.

© ENI Editions - All rigths reserved - Algeria Educ

assertTag()	assertNotTag()	Test de présence d'élément XML ou HTML.
assertThat()	assertNotThat()	Test de construction d'expression logique de test.
assertTrue()	assertNotTrue()	Test d'égalité avec la valeur Vrai.
assertType()	assertNotType()	Test d'égalité de type pour une variable.
assertXmlFileEqualsXmlFile()	assertNotXmlFileEqualsXmlFile()	Test d'équivalence entre deux contenus de fichiers XML.
assertXmlStringEqualsXmlFile ()	assertNotXmlStringEqualsXmlFile ()	Test d'équivalence entre un contenu XML et le contenu de fichier XML.
assertXmlStringEqualsXmlStr ()	assertNotXmlStringEqualsXmlStr()	Test d'équivalence entre deux contenus de chaînes XML.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction aux tests fonctionnels

Parmi tous les axes de qualité, nous pouvons citer :

- La pertinence des tests fonctionnels.
- La couverture des tests fonctionnels.
- Les temps de réponse.
- La stabilité en fonctionnement.
- La robustesse de l'application PHP.
- La pertinence des réponses.
- La robustesse du code face aux erreurs.
- La pertinence des messages d'erreur.

Le test fonctionnel est à l'application ce que le test unitaire est au code. Le rôle d'un test fonctionnel est de mettre en évidence le fonctionnement correct d'une fonctionnalité de l'application indépendamment du fonctionnement du code et de la manière dont l'application est réalisée techniquement.

Il est donc important de trouver un bon outil permettant de réaliser l'ensemble des étapes pour valider une fonctionnalité. Simple ou complexe, la fonctionnalité est avant tout un processus permettant de valider la réception d'une information attestant le bon fonctionnement de l'application.

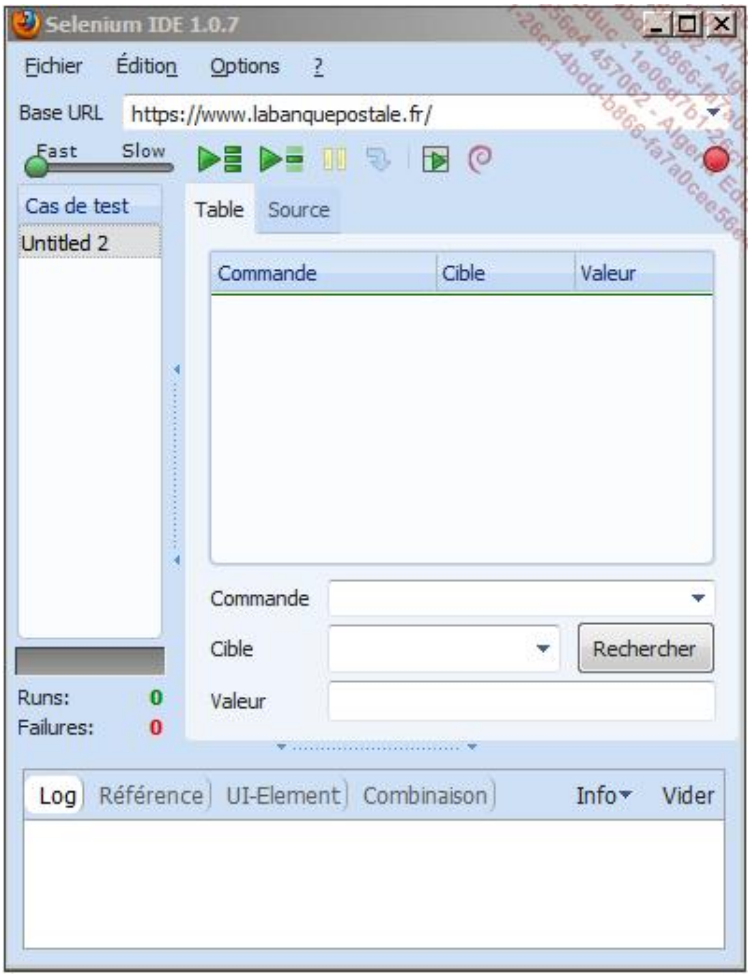
Parmi les bons outils permettant la validation et le test, SeleniumHq est un système de test des applications Web permettant d'automatiser les opérations de navigation au travers de Firefox. Essentiellement basée sur une puissante extension de Firefox, elle permet d'enregistrer des scénarios et de les rejouer à volonté.

Dès que votre jeu de tests est réalisé, il vous permet de valider les versions suivantes, de produire des plans de supervision réguliers et de réaliser un plan de tests réaliste ainsi qu'un plan de non-régression.

SeleniumHq est disponible à l'adresse suivante : http://seleniumhq.org/projects/ide/

L'essentiel de Selenium est une extension Firefox. Cependant, il possède l'ensemble des fonctionnalités pour le contrôler à distance et permettre une intégration totale au sein des environnements de tests automatisés et des environnements de développement.

© ENI Editions - All rigths reserved - Algeria Educ



L'enregistrement commence dès le clic sur l'icône d'enregistrement à droite.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction à la qualité de gestion des équipes

Parmi tous les axes de qualité, nous pouvons citer :

- La cohérence d'écriture du code.
- Les commentaires.
- La complexité du code.
- La pertinence des tests unitaires.
- La couverture des tests unitaires.
- La documentation des composants.
- La polyvalence des équipes de développement.

La qualité des équipes est, de loin, le plus complexe des sujets de ce chapitre.

En effet, il existe un nombre important de critères permettant de quantifier la qualité des équipes tels que :

- La charge de travail.
- L'environnement de travail.
- L'ambiance de travail entre les personnes.
- La qualité de l'encadrement.
- La motivation personnelle.
- La qualité des démarches de gestion des demandes.
- La diversité des tâches assignées aux développeurs.
- Le degré de liberté de réalisation des tâches.

Il est important de préserver un environnement qui mette à l'abri les équipes de développement du stress extérieur. En effet, plus les équipes seront sollicitées de toute part et de manière aléatoire, plus le stress et la chute de productivité seront importants.

Offrir un maximum de solutions techniques permettant d'isoler physiquement les équipes, proposer un outil de suivi d'anomalies type Mantis et un référentiel de source cohérent tel que subversion permettront de mieux gérer à distance l'importance de cette équipe.

Le but du manageur venant à encadrer un développement consiste à créer une véritable équipe dont les membres communiquent et partagent un ensemble de bonnes pratiques permettant à chaque membre d'être remplacé ou de venir en aide sur une autre partie des applications avec de très faibles efforts d'adaptation. Les environnements doivent, par conséquent, être le plus standard possible et ne pas laisser la possibilité de réaliser des tâches ordinaires sur le projet telles que :

- Installation de l'environnement de développement.
- Lancement des tests unitaires.
- Validation et centralisation du code.
- Validation du style du code.

© ENI Editions - All rigths reserved - Algeria Educ

- Structuration des fichiers de source PHP.
- Techniques de test du code.

© ENI Editions - All rigths reserved - Algeria Educ

Architecture simple et documentée

Une architecture logicielle est l'ossature primaire qui est représentée par :

- Des règles de nommage des différents éléments de l'architecture.
- Des règles de hiérarchisation des fichiers source.
- Des règles de codage et de formatage des scripts.
- Des règles de structure par composant.

Cette ossature va servir de base de réflexion à l'ensemble des développeurs de votre équipe, leur permettant d'entrer rapidement dans un composant développé par un autre membre de l'équipe.

Une architecture simple permet aussi de réduire la « courbe d'apprentissage » associée à chaque nouvelle entrée d'un développeur dans l'équipe.

Une architecture unifiée permet à chacun de retrouver rapidement :

- Le code source des classes métier.
- Le code des couches d'accès aux données et le code de persistance.
- Le code de test du composant.
- La documentation automatique et les outils de génération.
- Les outils permettant un environnement de développement rapide.

Pour cela, l'utilisation d'un gestionnaire de code source est souvent recommandée.

Voici quelques-uns des gestionnaires les plus répandus et utilisés :

- Subversion.
- CVS.
- GIT.
- Bazzar.
- Mercurial.
- Perforce.
- Visual SourceSafe.

1. Deux objectifs antagonistes

L'architecture logicielle au sens strict du terme doit permettre de concilier deux objectifs souvent antagonistes :

- Performance et qualité
- Adaptation et souplesse des développements

© ENI Editions - All rigths reserved - Algeria Educ

2. La qualité et la performance immédiate

L'impératif de qualité dans le monde Internet et du Web 2.0 est un enjeu majeur. Cette économie ne permet pas d'appliquer l'un des principes les plus fondamentaux de tout type de service : la qualité et le rendu d'un service proposé en ligne.

Appliqué à une application Web, ce principe va pousser les développeurs à fournir un code optimisé et souvent très spécifique, le rendant ainsi plus difficile à maintenir au sein d'un important référentiel de code. Effectivement, si un code simple (comme l'exemple suivant) est facilement maintenable en lui-même, la question de sa gestion devient problématique, voire ingérable, dans un ensemble de codes tous hétérogènes et spécifiques.

En effet, pour obtenir une performance et une qualité d'exception, un code épuré et optimisé peut être la réponse (tel le code étudié plus loin).

La première version du programme effectue l'ensemble des tests nécessaires à son fonctionnement sur tout type d'environnement. En effet, il est sage de valider les possibilités d'écriture d'un fichier ou d'ouverture de celui-ci.

```
<?php
$nomFichier = 'donnes.txt';
$contenu = $_POST['info'];

// Test de possibilité d'écriture du fichier
if (is_writable($nomFichier)) {
    echo "Pas de droit d'écriture pour le fichier ($filename)";
    exit;
}
if (!$hfichier = fopen($nomFichier, 'a')) {
    echo "impossible d'ouvrir le fichier ($nomFichier)";
    exit;
}

if (fwrite($hfichier, $contenu) === FALSE) {
    echo "Impossible d'écrire le fichier ($nomFichier)";
    exit;
}

fclose($hfichier);

?>
```

La deuxième version du programme limite son code au strict minimum afin de permettre la fonctionnalité.

L'ensemble des tests d'écriture repose sur la confiance envers le système d'exploitation et l'utilisation du script.

```
<?php
$nomFichier = 'donnees.txt';
$contenu = $_POST['info'];

$hfichier = fopen($filename, 'a');
fwrite($hfichier, $contenu);
fclose($hfichier);
?>
```

ou bien, encore plus simplement :

```
<?php
file_put_contents('donnees.txt', $_POST['info'], FILE_APPEND
);
?>
```

3. L'adaptation et la maintenance

Le premier impératif de pouvoir mener un projet important sur le long terme, sans dépendre des compétences d'un groupe limité d'individus, est totalement justifié.

Le second impératif est la maîtrise de la qualité et des performances dans le temps et ceci en permettant l'extension et la modification du code source durant le cycle de vie des applications et des projets.

© ENI Editions - All rigths reserved - Algeria Educ

Pour cela, il faut souvent sacrifier la performance en passant par des frameworks permettant : d'unifier le code au sein d'un même modèle, de contrôler finement les règles de test, de routage, d'accès aux pages.

En utilisant un framework tel que Zend Framework, avant d'exécuter le code relatif à l'écriture des données postées dans votre fichier, la couche applicative devra effectuer les opérations suivantes :

1. Analyse de la requête.
2. Création d'un objet contenant les informations de la requête.
3. Recherche des informations sur les traitements à réaliser.
4. Création d'un objet contenant les informations de la réponse.
5. Application de traitement avant l'exécution des actions.
6. Appel des différentes actions à réaliser.
7. Application de traitement après l'exécution des actions.
8. Renvoi de la réponse.

Il est donc évident dans cet environnement que l'exécution ne peut être plus rapide. Cependant, le framework pourra évoluer plus sereinement en ne permettant pas l'amalgame au sein d'une même application de plusieurs codes PHP ayant chacun une architecture spécifique.

Unir ses forces autour d'une architecture commune est préférable dans le temps ; cette architecture doit allier la qualité, la performance et surtout l'homogénéité des approches de chaque fonctionnalité d'une application PHP.

© ENI Editions - All rigths reserved - Algeria Educ

Découpage en couches : le modèle MVC2

Le Framework Zend prône une approche par couche implémentant le modèle MVC2.

Ce modèle propose de découper les applications ayant un rendu visuel en trois couches ayant chacune un rôle spécifique et identifié :

- La couche Model ou Modèle
- La couche View ou Présentation
- La couche Control ou Contrôle

1. La couche Modèle

Cette couche a pour but de contrôler la structure de vos données dans les zones de persistance (fichier, base de données, XML, etc.) ainsi que de définir vos modèles de classe et l'ensemble des informations concernant vos données et leur organisation.

Cette couche doit au minimum offrir l'ensemble des fonctionnalités proposées par l'approche CRUD et permettant les opérations suivantes :

- Création d'objet PHP (Create)
- Lecture d'objet PHP (Read)
- Mise à jour d'objet PHP (Update)
- Suppression d'objet PHP (Delete)

2. La couche Présentation

Cette couche a pour but de gérer le rendu visuel de l'application. Dans le cas d'une application Web, les moteurs de modèle (template) sont particulièrement adaptés à ce type d'environnement et permettent d'offrir des fonctionnalités avancées telles que la gestion des caches et de la performance.

Les moteurs de modèle disposent d'un langage très simple permettant de créer rapidement un look spécifique pour une application et d'obtenir des applications à thèmes, adaptables au choix de chaque utilisateur.

Parmi les moteurs de modèle, il est impératif de citer le projet Smarty (http///www.smarty.net/).

3. La couche Contrôle

Cette couche a pour but de gérer les aspects les plus fondamentaux des traitements dits « métier » de l'application.

Ses deux rôles essentiels sont de contenir les traitements fondamentaux et de garantir la bonne coordination entre la gestion du rendu visuel (couche Présentation) et la gestion des données (couche Modèle).

Cette couche est la plus importante car elle porte l'ensemble des traitements de l'application !

© ENI Editions - All rigths reserved - Algeria Educ

Importance de la modélisation UML

Tout projet souhaitant progresser se doit d'avoir une architecture d'objet structurée et modélisée correctement.

Aucun projet ne peut dépasser ses limites de performance, de qualité et de fonctionnalités sans planification préalable. Cette planification, c'est la modélisation UML.

1. Pourquoi la modélisation UML ?

La modélisation UML est un standard et un consensus très largement répandu dans l'ensemble des industries de communication, des industriels et des concepteurs de logiciels tous secteurs confondus. L'OMG (*Object Management Group*) est un organisme qui offre une structure pour maintenir et faire évoluer l'UML depuis sa première version 1.0 en 1991.

Les besoins des utilisateurs sont mis au centre de la modélisation UML. Les choix de conception seront donc toujours fortement influencés par les besoins des utilisateurs.

2. L'approche MDA

L'approche MDA ou approche pilotée par l'architecture est une approche prônant la domination des modèles UML sur le code.

C'est-à-dire que la source qui fait foi en cas de litige n'est plus le code mais bien le modèle. Cela implique que toute modification doit passer par une retouche plus ou moins importante des modèles UML. Puis ces modifications doivent être réintroduites dans le code, soit de manière manuelle dans le pire des cas, soit semi-manuellement ou automatiquement. Cela dépend grandement de votre environnement et de votre plate-forme technique.

Dans la plupart des cas, la traduction touche deux types de diagrammes importants dans la modélisation UML : les diagrammes de classe et les diagrammes de séquence.

Voici quelques exemples de traduction manuelle de diagramme UML en PHP.

3. Translation diagramme de classe UML et langage technique

a. Introduction à la traduction de modèle UML

La traduction d'UML vers le code source s'appuie sur deux points importants :

- Les stéréotypes qui sont les annotations entre jalons << >>.
- Le profil utilisé par le générateur qui est un ensemble de modèles et de règles de traduction, spécifique à votre projet.

Selon les profils de génération, il est possible de produire un ou plusieurs fichiers relatifs à chaque classe de votre modèle.

Rien ne vous empêche de générer plusieurs fichiers pour chaque classe de manière automatique !

Nous allons donc voir deux traductions possibles d'un diagramme de classe en code PHP et en langage SQL afin de garantir une base de persistance pour nos données entre différents appels de l'application.

Notre exemple ne prend pas en compte la génération d'une classe de translation objet PHP / SQL appelée par AAO (*Adaptateur d'Accès aux Objets*).

b. Traduction de modèle : processus imparfait

Une traduction de modèle entraîne, à chaque étape, une déformation (voire une perte d'information) lors de sa conversion dans un langage de programmation, dialecte technique ou documentation quelconque.

© ENI Editions - All rigths reserved - Algeria Educ

En effet, il existe de nombreuses limitations dans les langages techniques ne permettant pas de supporter l'intégralité des concepts objet offerts par le langage UML et son méta-modèle.

Ainsi il est difficile de traduire des concepts d'interface UML en langage SQL alors que cela paraît d'une simplicité et d'une cohérence évidente en langage PHP orienté objet. En effet, une interface UML trouvera sa traduction naturelle en interface PHP car le concept objet d'interface est le même entre UML et le PHP.

Un autre problème de taille est la traduction des types UML en PHP !

Le langage PHP étant un langage d'inférence, c'est-à-dire un langage dont le type de chaque variable est défini au moment de l'affectation de valeur, le passage du type dans la traduction en PHP ne peut être que l'objet d'un compromis.

Une des solutions consiste à poser systématiquement un commentaire indiquant précisément le type de donnée attendue dans chaque attribut de la classe.

Cette traduction est, certes, imparfaite mais c'est le propre de toute traduction.

Pour anecdote, la traduction UML s'apparente parfois à la traduction japonaise d'un roman français parlant de Berlin. Le lieu populaire de Berlin doit-il être traduit par un autre lieu populaire de Corée du Sud ou doit-on faire une traduction purement « technique ». Ou bien encore, doit-on produire une traduction contextualisée ? Ce problème n'est pas simple à résoudre car sa réponse est relative à l'environnement du projet et de ses différentes contraintes.

Ce que cet ouvrage peut cependant vous proposer, c'est de mettre en évidence les relations qu'il peut y avoir entre un diagramme UML, du code PHP et le langage SQL. L'adaptation de la traduction reste entièrement possible dans tous les cas.

Dans la première partie de l'exemple, la traduction de diagramme UML en langage PHP est abordée, ainsi que la génération de classes de test pour PHPUnit et la mise en place des bases de documentation pour phpDocumentor afin de faciliter la génération de la documentation du projet.

c. Traduction de modèle : outil imparfait

De nombreux outils existent pour la traduction de modèle. Cependant, les traducteurs fournis par les ateliers de modélisation UML en source libre tels que StarUML ou argoUML ne produisent qu'un code ne permettant pas une traduction fine des modèles produits pour vos projets.

L'expérience d'une traduction manuelle de votre conception, au moins une fois dans votre carrière de développeur PHP en environnement de conception UML orienté objet, vous permettra de mieux comprendre ce que des changements dans une classe introduisent comme changement dans les modèles UML et inversement.

© ENI Editions - All rigths reserved - Algeria Educ

Translation diagramme de classe UML vers PHP objet

L'ensemble du processus de traduction présent dans ce chapitre permet de traduire des diagrammes UML directement en code PHP.

Il est impératif de bien maîtriser ce processus et donc de comprendre comment une classe UML peut se traduire en PHP.

Le code proposé comme traduction n'est pas une vérité absolue mais plutôt une solution reflétant au mieux le diagramme UML.

1. Traduction d'une classe vide en PHP

Prenons une classe simple : Utilisateur. Voici donc sa représentation UML :

Un générateur UML avec un profil très simple pourra générer le code suivant :

```
<?php
class Utilisateur {

}
?>
```

Un générateur UML avec un profil moins trivial pourra générer plusieurs codes tels que :

- Le code source pré-documenté pour phpDocumentor.
- La classe de test sur la base de PHPUnit.
- Le code SQL de création de table.
- Le code basique d'une classe permettant d'accéder aux objets via la base de données.

Voici ce qu'il pourra générer comme code PHP et comme classe de test.

a. Traduction en classe PHP avec documentation

Ici, le processus de traduction est clair et simple. Une classe UML équivaut à une classe PHP.

```
<?php
/**
* Classe Utilisateur
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Utilisateur {

}
?>
```

b. Traduction en classe PHP de test PHPUnit

© ENI Editions - All rigths reserved - Algeria Educ

La classe de test est plus rudimentaire et peut être générée afin de permettre de faciliter la réalisation du code de tests unitaires basiques.

Ici, les tests sont très simples :

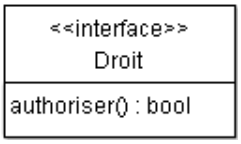
- Création d'objet.
- Validation du type de l'objet.
- Validation de la non-nullité de l'objet.

```
<?php
require_once 'PHPUnit/Framework.php';
require_once 'src/base/Utilisateur.class.php';

/**
* Classe TestUtilisateur
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class TestUtilisateur extends PHPUnit_Framework_TestSuite
{
/**
* @var objet Utilisateur
* @access private
**/
private $util1;

/**
* initialise l'objet Utilisateur
* @access protected
**/
protected function setUp()
    {
        $this->util1=new Utilisateur();
    }

/**
* libère l'objet Utilisateur
* @access protected
**/
protected function tearDown()
    {
        $this->util1 = NULL;
    }


/**
* teste la bonne création de l'utilisateur
* @access public
**/
public function testCreationUtilisateur()
    {

    $this->assertNotNull( $this->util1);
    }

/**
* teste le type de l'utilisateur
* @access public
**/
public function testObjetUtilisateur()
    {
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    $this->assertType('object', $this->util1);
    }
}

?>
```

2. Traduction d'une interface



Voici ce qui pourra être généré comme code PHP et comme classe de test.

a. Traduction en classe PHP avec documentation

Depuis la version 5 de PHP, un véritable langage objet est apparu, introduisant une nouvelle syntaxe pour déclarer des interfaces. Voici donc une traduction plus naturelle de l'interface en UML.

```
<?php
/**
* Interface Droit
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
interface Droit {
    public function authoriser() ;
}
?>
```

Une seconde traduction peut avoir lieu, il s'agit de traduire l'interface en classe abstraite.

```
<?php
/**
* classe abstraite pour Droit
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
abstract class Droit {
    public abstract function authoriser() ;
}
?>
```

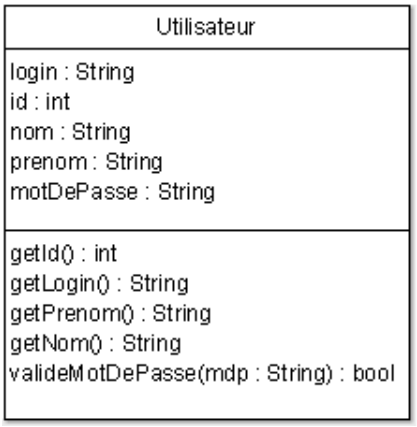
b. Traduction en classe PHP de test PHPUnit

Le générateur ne pourra pas générer automatiquement de code de test de bonne qualité pour une interface.

En effet, une interface n'a de sens que lorsqu'une classe concrète réalise ou implémente les méthodes qu'expose l'interface.

3. Traduction d'une classe et de ses propriétés

© ENI Editions - All rigths reserved - Algeria Educ



a. Traduction en classe PHP avec documentation

Toute la difficulté est maintenant de traduire directement les propriétés UML en variables et méthodes de classe.

Ce qui est proposé dans cette traduction est le respect de l'encapsulation en déclarant tous les attributs privés et en déclarant des méthodes d'accès aux attributs (ou accesseurs).

De plus, la traduction ajoute automatiquement un constructeur de classe PHP permettant la création des objets avec les valeurs des attributs en paramètre.

```
<?php
/**
* Classe Utilisateur
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Utilisateur {
    /**
    * @access private
    * @var string
    */
    private $login;

    /**
    * @access private
    * @var integrer
    */
    private $id;

    /**
    * @access private
    * @var string
    */
    private $nom;

    /**
    * @access private
    * @var string
    */
    private $prenom;

    /**
    * @access private
    * @var string
    */
    private $motDePasse;
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    /**
    * @access public
    * @param none
    * @return string
    */
    public function getLogin() {
        return $this->login;
    }

    /**
    * @access public
    * @param none
    * @return integer
    */
    public function getId() {
        return $this->id;
    }
    /**
    * @access public
    * @param none
    * @return string
    */
    public function getPrenom() {
        return $this->login;
    }
    /**
    * @access public
    * @param none
    * @return string
    */
    public function getNom() {
        return $this->nom;
    }
    /**
    * @access public
    * @param string
    * @return boolean
    */
    public function valideMotDePasse ($mdp) {
        $retour=FALSE;
        # Code à réaliser
        return $retour;
    }

    /**
    * Contructeur
    * @access public
    */
    public function __construct(     $login="",
                                     $id=0,
                                     $nom="",
                                     $prenom="",
                                     $motDePasse="")
    {
        $this->login=$login;
        $this->id=$id;
        $this->nom=$id;
        $this->prenom=$prenom;
        $this->motDePasse=$motDePasse;
    }
}
?>
```

b. Traduction en classe PHP de test PHPUnit

Le code produit les mêmes tests de non-nullité et ajoute les tests d'appel à l'ensemble des méthodes de l'objet.

© ENI Editions - All rigths reserved - Algeria Educ

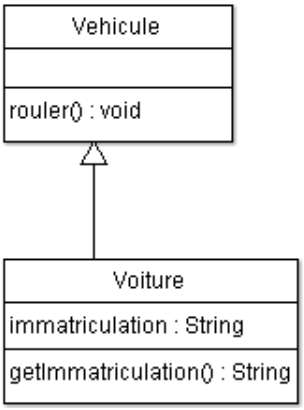
```
<?php
require_once 'PHPUnit/Framework.php';
require_once 'src/base/Utilisateur.class.php';

/**
* Classe TestUtilisateur
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class TestUtilisateur extends PHPUnit_Framework_TestSuite
{
/**
* @var objet Utilisateur
* @access private
**/
private $util1;

/**
* @var objet Utilisateur
* @access private
**/
private $util2;

/**
* initialise l'objet Utilisateur
* @access protected
**/
protected function setUp()
    {
        $this->util1=new Utilisateur();
    }

/**
* libère l'objet Utilisateur
* @access protected
**/
protected function tearDown()
    {
        $this->util1 = NULL;
    }
/**
* teste la bonne création de l'utilisateur
* @access public
**/
public function testCreationUtilisateur()
    {

    $this->assertNotNull( $this->util1);
    }
/**
* teste le type de l'utilisateur
* @access public
**/
public function testObjetUtilisateur()
    {

    $this->assertType('object', $this->util1);
    }
/**
* teste la récupération de Login de la classe Utilisateur
* @access public
**/
public function testGetLoginUtilisateur()
    {
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    $this->assertNotNull( $this->util1->getLogin());
    }


/**
* teste la récupération de Id de la classe Utilisateur
* @access public
**/
public function testGetIdUtilisateur()
    {

    $this->assertNotNull( $this->util1->getId());
    }

/**
* teste la récupération de Nom de la classe Utilisateur
* @access public
**/
public function testGetNomUtilisateur()
    {

    $this->assertNotNull( $this->util1->getNom());
    }

/**
* teste la récupération de Prenom de la classe Utilisateur
* @access public
**/
public function testGetPrenomUtilisateur()
    {

    $this->assertNotNull( $this->util1->getPrenom());
    }

/**
* teste la méthode valideMotDePasse de la classe Utilisateur
* @access public
**/
public function testValideMotDePasseUtilisateur()
    {

    $this->assertTrue ( $this->util1->valideMotDePasse(""));
    # ou bien
    $this->assertFalse ( $this->util1->valideMotDePasse(""));
    }

}

?>
```

4. Traduction de l’héritage en classe

© ENI Editions - All rigths reserved - Algeria Educ



a. Traduction en classe PHP Véhicule

```
<?php
/**
* Classe Vehicule
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Vehicule {
    /**
    * @access public
    * @param none
    * @return void
    */
    public function rouler() {
    }

    /**
    * Constructeur
    * @access public
    */
    public function __construct()
    {
    }
}
?>
```

b. Traduction en classe de test PHPUnit pour Véhicule

```
<?php
require_once 'PHPUnit/Framework.php';
require_once 'src/base/Utilisateur.class.php';

/**
* Classe TestVehicule
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class TestVehicule extends PHPUnit_Framework_TestSuite
{
/**
```


© ENI Editions - All rigths reserved - Algeria Educ

```
* @var objet Vehicule
* @access private
**/
private $vehcule1;

/**
* initialise l'objet Vehicule
* @access protected
**/
protected function setUp()
    {
        $this->vehicule1=new Vehicule();
    }

/**
* libère l'objet Vehicule
* @access protected
**/
protected function tearDown()
    {
        $this-> vehicule1 = NULL;
    }


/**
* teste la bonne création de vehicule
* @access public
**/
public function testCreationVehicule()
    {

    $this->assertNotNull( $this-> vehicule1);
    }

/**
* teste le type de Vehicule
* @access public
**/
public function testObjetVehicule()
    {

    $this->assertType('object', $this->vehicule1);
    }
}
?>
```

c. Traduction en classe PHP Voiture

```
<?php
/**
* Classe Voiture
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Voiture extends Vehicule {
    /**
    * @access private
    * @var string
    */
    private $immatriculation;

    /**
    * @access public
```

© ENI Editions - All rigths reserved - Algeria Educ

```
    * @param none
    * @return string
    */
    public function getImmatriculation() {
        return $this->immatriculation;
    }

    /**
    * Constructeur
    * @access public
    */
    public function __construct($immatriculation="" )
    {
        $this->immatriculation =$immatriculation;
    }
}
?>
```

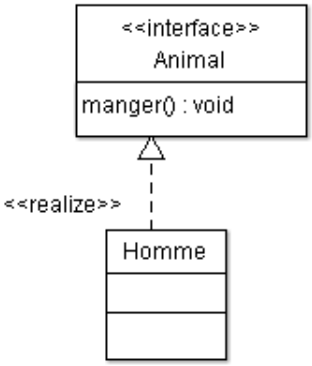
d. Traduction en classe de test PHPUnit pour Voiture

```
<?php
require_once 'PHPUnit/Framework.php';
require_once 'src/base/Utilisateur.class.php';

/**
* Classe TestVoiture
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class TestVoiture extends PHPUnit_Framework_TestSuite
{
/**
* @var objet Voiture
* @access private
**/
private $voiture;

/**
* @var objet Voiture
* @access private
**/
private $voiture;

/**
* initialise l'objet Voiture
* @access protected
**/
protected function setUp()
    {
        $this->voiture=new Voiture ();
    }

/**
* libère l'objet Voiture
* @access protected
**/
protected function tearDown()
    {
        $this->voiture = NULL;
    }


/**
```


© ENI Editions - All rigths reserved - Algeria Educ

```
* teste la bonne création de voiture
* @access public
**/
public function testCreationVoiture()
    {

    $this->assertNotNull( $this->voiture);
    }

/**
* teste le type de voiture
* @access public
**/
public function testObjetVoiture ()
    {

    $this->assertType('object', $this->voiture);
    }


/**
* teste la récupération de Immatriculation de la classe Voiture
* @access public
**/
public function testGetImmatriculationVoiture ()
    {

    $this->assertNotNull( $this->voiture->getImmatriculation());
    }
}

?>
```

5. Traduction de l'héritage d'interface



a. Traduction en interface PHP avec documentation

```
<?php
/**
* Interface Animal
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
interface Animal {
    public function manger() ;
}
```

© ENI Editions - All rigths reserved - Algeria Educ

```
?>
```

b. Traduction en classe PHP Homme

```
<?php
/**
* Classe Homme
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Homme implements Animal {
    /**
    * @access public
    * @param none
    * @return string
    */
    public function manger(){
    }

    /**
    * Constructeur
    * @access public
    */
    public function __construct()
    {
    }
}
?>
```

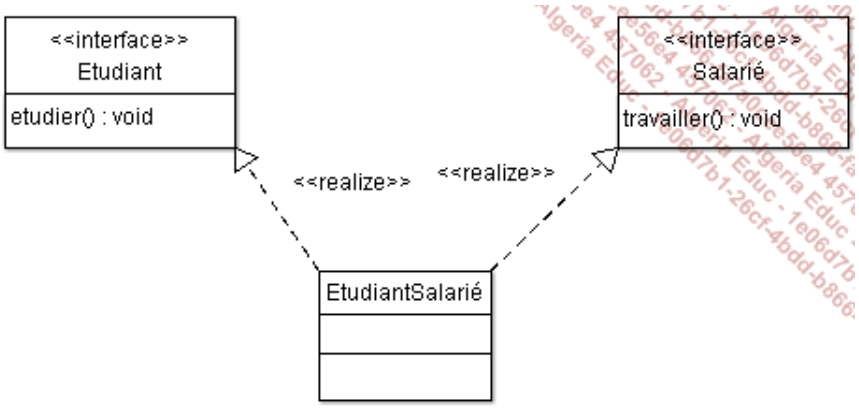
c. Traduction en classe PHP de test PHPUnit

```
<?php
require_once 'PHPUnit/Framework.php';
require_once 'src/base/Homme.class.php';

/**
* Classe TestHomme
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class TestHomme extends PHPUnit_Framework_TestSuite
{

/**
* @var objet Homme
* @access private
**/
private $homme;

/**
* initialise l'objet Homme
* @access protected
**/
protected function setUp()
    {
        $this->homme=new Homme ();
    }
```


© ENI Editions - All rigths reserved - Algeria Educ

```
/**
* libère l'objet Homme
* @access protected
**/
protected function tearDown()
    {
        $this->homme = NULL;
    }


/**
* teste la bonne création de homme
* @access public
**/
public function testCreationHomme ()
    {

    $this->assertNotNull( $this->homme);
    }

/**
* teste le type de homme
* @access public
**/
public function testObjetHomme ()
    {

    $this->assertType('object', $this->homme);
    }


/**
* teste la méthode manger de la classe Homme
* @access public
**/
public function testMangerHomme ()
    {
    $this->homme->manger();
    }
}

?>
```

6. Traduction de l'héritage d'interface multiple



a. Traduction en interface PHP Etudiant

```
<?php
```

```
/**
* Interface Etudiant
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
interface Etudiant {
    public function etudier() ;
}
?>
```

b. Traduction en interface PHP Salarie

```
<?php
/**
* Interface Salarie
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
interface Salarie {
     public function travailler() ;
}
?>
```

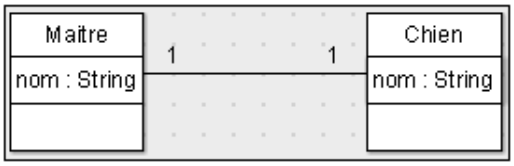
c. Traduction en interface PHP EtudiantSalarie

```
<?php
/**
* Classe EtudiantSalarie
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class EtudiantSalarie implements Etudiant, Salarie {
    /**
    * @access public
    * @param none
    * @return none
    */
    public function etudier(){
    }

    /**
    * @access public
    * @param none
    * @return none
    */
    public function travailler(){
    }

    /**
    * Constructeur
    * @access public
    */
    public function __construct()
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    {
    }
}
?>
```

7. Traduction d'une association 1-1



a. Traduction en classe PHP Maître

```
<?php
/**
* Classe Maitre
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Maitre {
    /**
    * @access private
    * @var objet Chien
    */
    private $monChien;
    /**
    * @access private
    * @var string
    */
    private $nom;

    /**
    * @access public
    */
    public function __construct($nom, $chien=null) {
        $this->nom=$nom;
        if (isset($chien)) {
            $this->monChien=$chien;
            $chien->setMaitre($this);
        }
    }

    /**
    * @access public
    */
    public function __toString() {
        #var_dump($this);

        $str="Maitre [ Nom: ".$this->nom."]";
        if (isset($this->monChien)) {
            $str.="\nMon chien : ".$this->monChien->getNom();
        } else { $str.="\nPas de chien, je ne suis pas un maître-
chien"; }
        $str.="\n\n";
        return $str;
    }
    /**
    * @access public
```

© ENI Editions - All rigths reserved - Algeria Educ

```
    */
    public function setChien($c) {
        $this->monChien=$c;
    }

    /**
    * @access public
    */
    public function getNom() { return $this->nom; }
}
?>
```

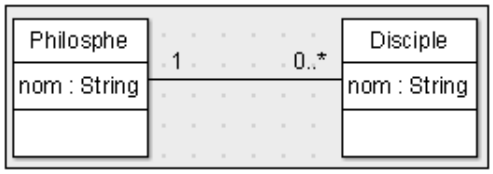
b. Traduction en classe PHP Chien

```
<?php

/**
* Classe chien
* Date : 27/01/2010 23 :05
* @author Jean-Marie Renouard
* @ copyright ENI 2010
* @licence GNU GPL 3.0
* @version 0.1
* @package base
*/
class Chien {
    /**
    * @access private
    * @var objet Maitre
    */
    private $monMaitre;
    /**
    * @access private
    * @var string
    */
    private $nom;

    /**
    * @access public
    */
    public function __construct($nom, $Maitre=null) {
        $this->nom=$nom;
        if (isset($Maitre)) {
            $this->monMaitre=$Maitre;
            $Maitre->setChien($this);
        }
    }

    /**
    * @access public
    */
    public function __toString() {
        $str="Chien [ Nom: ".$this->nom."]";
        if (isset($this->monMaitre)) {
            $str.="\nMon Maitre : ".$this->monMaitre-
>getNom();
        } else { $str.="\nPas de maître, Chien errant."; }
        $str.="\n";
        return $str;
    }

    /**
    * @access public
    */
    public function setMaitre($m) {
        $this->monMaitre=$m;
    }
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    public function getNom() { return $this->nom; }
}
?>
```

8. Traduction d'une association 1-N



a. Traduction en classe PHP Philosophe

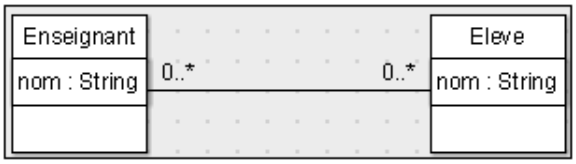
```
<?php

class Philosophe {
    private $disciples;
    private $nom;

    public function __construct($nom, $etuds=null) {
        $this->nom=$nom;

        $this->disciples=array();

        if (isset($etuds)) {
            if ( is_array($etuds) ) {
                   $this->disciples=array_unique($etuds);
            } else {
                   array_push($this->disciples, $etuds);
            }
        }

        foreach ($this->disciples as $etu) { $etu-
>setPhilosophe($this); }
    }

    public function __toString() {
        $str="Philosophe [ Nom: ".$this->nom."]";
        if (count($this->disciples) > 0) {
            $str.="\n\ts'occupe des disciples :";
            foreach ($this->disciples as $etu) {
$str.="\n\t\t * ". $etu->getNom(); }
        } else { $str.="\n\tPas de disciple."; }
        $str.="\n";
        return $str;
    }

    public function addDisciple($etu, $propage=true) {
        if (!in_array($etu, $this->disciples)) array_push($this-
>disciples, $etu);
        if ($propage) $etu->addPhilosophe($this, false);
    }

    public function delDisciple($etu, $propage=true) {
        $this->disciples= array_diff($this->disciples,
array($etu));
        if ($propage) $etu->delPhilosophe($this, false);
    }
    public function getNom() { return $this->nom; }
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

b. Traduction en classe PHP Disciple

```
<?php

class Disciple {
    private $philosophe;
    private $nom;

    public function __construct($nom, $phil=null) {
        $this->nom=$nom;

        $this->philosophe=$phil;

    }

    public function __toString() {
        $str="Disciple [ Nom: ".$this->nom."\n\tPhilosophe:
".$this->philosophe->getNom()."]";
        $str.="\n";
        return $str;
    }

    public function setPhilosophe($phil) {
        $this->philosophe=$phil;
    }

    public function getPhilosophe() {
        return $this->philosophe;
    }
}
?>
```

9. Traduction d'une association N-N



a. Traduction en classe PHP Enseignant

```
<?php

class Enseignant {
    private $etudiants;
    private $nom;

    public function __construct($nom, $etuds=null) {
        $this->nom=$nom;

        $this->etudiants=array();

        if (isset($etuds)) {
            if ( is_array($etuds) ) {
                $this->etudiants=array_unique($etuds);
            } else {
                array_push($this->etudiants, $etuds);
            }
        }

        foreach ($this->etudiants as $etu) { $etu-
```


© ENI Editions - All rigths reserved - Algeria Educ

```
>addEnseignant($this); }
    }

    public function __destruct() {
        foreach ($this->etudiants as $etu) { $etu-
>delEnseignant($this); }
    }


    public function __toString() {
        $str="Enseignant [ Nom: ".$this->nom."]";
        if (count($this->etudiants) > 0) {
            $str.="\n\ts'occupe des étudiants :";
            foreach ($this->etudiants as $etu) {
$str.="\n\t\t * ". $etu->getNom(); }
        } else { $str.="\n\tPas d etudiant."; }
        $str.="\n";
        return $str;
    }

    public function addEtudiant($etu, $propage=true) {
        if (!in_array($etu, $this->etudiants)) array_push($this-
>etudiants, $etu);
        if ($propage) $etu->addEnseignant($this, false);
    }

    public function delEtudiant($etu, $propage=true) {
        $this->etudiants= array_diff($this->etudiants,
array($etu));
        if ($propage) $etu->delEnseignant($this, false);
    }
    public function getNom() { return $this->nom; }
}
?>
```

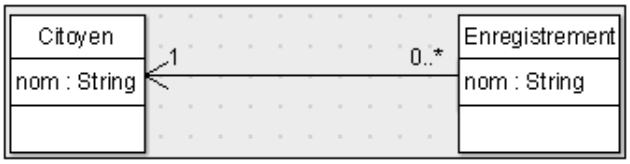
b. Traduction en classe PHP Etudiant

```
<?php

class Etudiant {
    private $enseignants;
    private $nom;

    public function __construct($nom, $enss=null) {
        $this->nom=$nom;

        $this->enseignants=array();

        if (isset($enss)) {
            if ( is_array($enss) ) {
                $this->enseignants=array_unique($enss);
            } else {
                array_push($this->enseignants, $enss);
            }
        }

        foreach ($this->enseignants as $ens) { $ens-
>addEtudiant($this); }
    }

    public function __destruct() {
        foreach ($this->enseignants as $ens) { $ens-
>delEtudiant($this); }
    }


    public function __toString() {
```

© ENI Editions - All rigths reserved - Algeria Educ

```
        $str="Etudiant [ Nom: ".$this->nom."]";
        if (count($this->enseignants) > 0) {
            $str.="\n\tsuit les cours de :";
            foreach ($this->enseignants as $ens) {
$str.="\n\t\t * ". $ens->getNom(); }
        } else { $str.="\n\tPas d enseignant."; }
        $str.="\n";
        return $str;
    }

    public function addEnseignant($ens, $propage=true) {
        if (!in_array($ens, $this->enseignants))
array_push($this->enseignants, $ens);
        if ($propage) $ens->addEtudiant($this, false);
    }

    public function delEnseignant($ens, $propage=true) {
        $this->enseignants= array_diff($this->enseignants,
array($ens));
        if ($propage) $ens->delEtudiant($this, false);
    }
    public function getNom() { return $this->nom; }
}
?>
```

10. Traduction d'une association 1-N à navigation restreinte



Dans cet exemple le code est similaire au code du philosophe et des disciples. Cependant le code sera allégé côté de la classe Citoyen où aucune information sur les enregistrements ne sera stockée dans un tableau PHP (pas d'attribut enregistrements). Dans le cas d'une navigation restreinte, le code produit est plus simple.

Le fait de poser des navigations restreintes dans votre code permet de simplifier le code et donc d'offrir plus de performance en exécution car moins de code est nécessaire pour créer un objet. Le stockage en base de données ou sous forme sérialisé dans un fichier ou en mémoire, par exemple, prendra moins d'espace et donc fera un bon candidat pour la montée en charge.

a. Traduction en classe PHP Citoyen

```
<?php

class Citoyen {
    private $nom;

    public function __construct($nom) {
        $this->nom=$nom;


    public function __toString() {
        $str="Citoyen [ Nom: ".$this->nom."]";
        $str.="\n";
        return $str;
    }

    public function getNom() { return $this->nom; }
}
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

b. Traduction en classe PHP Enregistrement

```
<?php
class Enregistrement {
    private $citoyen;
    private $nom;

    public function __construct($nom, $cito=null) {
        $this->nom=$nom;

        $this->citoyen=$cito;

    }

    public function __toString() {
        $str="Enregistrement [ Nom: ".$this->nom."\n\tCitoyen:
".$this->citoyeni->getNom()."]";
        $str.="\n";
        return $str;
    }

    public function setCitoyen($cito) {
        $this->citoyen=$cito;
    }

    public function getCitoyen() {
        return $this->citoyen;
    }
}
?>
```

11. Traduction d'agrégations

Le langage PHP n'offre que très peu de moyens de traduire une distinction forte entre une association et une agrégation et nous nous contenterons de traduire ce lien d'association fort comme nous avons procédé avec les associations. La seule chose que nous pouvons réaliser est la mise en place de commentaires spécifiques dans le code indiquant le lien fort mis en évidence dans la modélisation du système.

© ENI Editions - All rigths reserved - Algeria Educ

Translation diagramme de classe UML vers langage SQL

1. Traduction des différents types de classe

Même si la classe UML Utilisateur ne possède aucun attribut, lors de la génération de la table UTILISATEUR qui sera l'équivalent au sens SQL de cette classe, il est impératif d'ajouter un identifiant unique ID_UTILISATEUR qui permettra de jouer le rôle d'identifiant d'objet dans notre modèle.

Ainsi, chaque objet de type classe Utilisateur aura un équivalent au sens strict du terme en base de données.

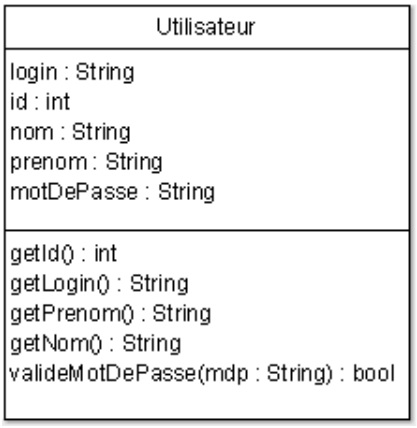
a. Traduction en code SQL

```
CREATE TABLE 'base'.'UTILISATEUR' (
'ID_UTILISATEUR' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
```

2. Traduction d'une interface UML

Une interface pose un problème fondamental de traduction. En effet, l'interface ne porte pas d'information en soi et n'existe qu'au travers de classes de réalisation (souvent des classes concrètes) traduites elles-mêmes en SQL par des tables comportant déjà des colonnes d'information relatives à chaque attribut de la classe.

3. Traduction d'une classe et de ses propriétés



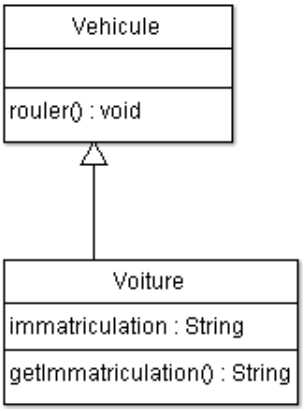
a. Traduction en code SQL

Le processus de traduction ici épure grandement la sémantique UML car il est impossible de traduire en SQL des fonctions telles que les accesseurs.

```
CREATE TABLE 'base'.'UTILISATEUR' (
'ID_UTILISATEUR' INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
'LOGIN' VARCHAR(255),
'NOM' VARCHAR(255),
'PRENOM' VARCHAR(255),
'MOTDEPASSE' VARCHAR(255)
);
```

© ENI Editions - All rigths reserved - Algeria Educ

4. Traduction de l'héritage de classe



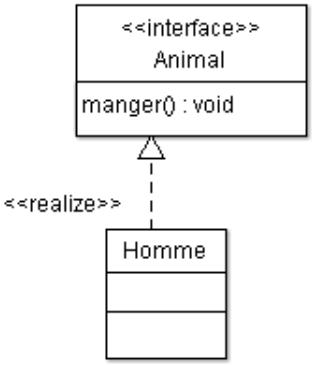
a. Traduction en code SQL classe mère

```
CREATE TABLE 'base'.'VEHICULE' (
'ID_VEHICULE' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
```

b. Traduction en code SQL classe fille

```
CREATE TABLE 'base'.'VOITURE' (
'ID_VOITURE' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'ID_VEHICULE' INT NOT NULL,
'IMMATRICULATION'VARCHAR(255),
FOREIGN KEY ('ID_VEHICULE') REFERENCES 'base'.'VEHICULE'
('ID_VEHICULE') ON DELETE CASCADE
);
```

5. Traduction de l'héritage d'interface



a. Traduction en code SQL classe fille

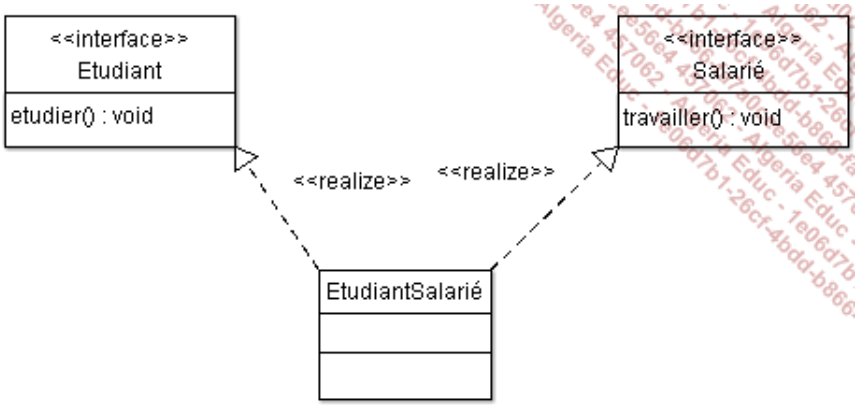
Il n'est pas possible de traduire en SQL le résultat d'une interface car une interface n'a pas d'identité propre. La solution consiste donc à ajouter un champ indiquant les interfaces implémentées sans créer de table pour l'interface.

```
CREATE TABLE 'base'.'HOMME' (
```


© ENI Editions - All rigths reserved - Algeria Educ

```
'ID_HOMME' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'INTERFACES' VARCHAR(255) NOT NULL
);
```

6. Traduction de l'héritage d'interface multiple



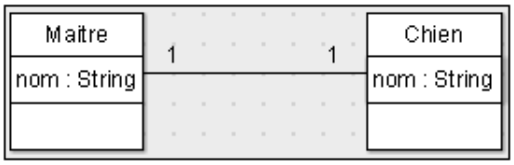
a. Traduction en code SQL classe fille

Comme dans l'exemple précédent, un champ avec le nom des interfaces séparées par une virgule permet l'implémentation de la notion d'héritage.

```
CREATE TABLE 'base'.'ETUDIANTSALARIE' (
'ID_ ETUDIANTSALARIE ' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'INTERFACES' VARCHAR(255) NOT NULL
);
```

Pour des raisons de non-existence d'une interface, nous n'avons pas créé de table définissant les interfaces. Il s'agit d'un choix conceptuel. Cependant, l'implémentation des interfaces sous forme de tables ou de vues SQL est totalement possible.

7. Traduction d'une association 1-1



a. Traduction de la classe Maître en code SQL

```
CREATE TABLE 'base'.'MAITRE' (
'ID_MAITRE' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'ID_CHIEN' INT NOT NULL,
FOREIGN KEY ('ID_CHIEN') REFERENCES 'base'.'CHIEN'
('ID_CHIEN') ON DELETE CASCADE
);
```

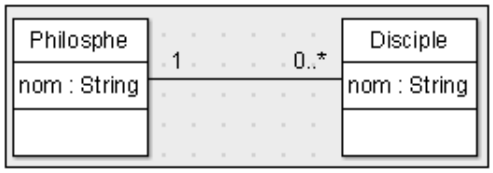
b. Traduction de la classe Chien en code SQL

```
CREATE TABLE 'base'.'CHIEN' (
'ID_CHIEN' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'ID_MAITRE' INT NOT NULL,
```

© ENI Editions - All rigths reserved - Algeria Educ

```
FOREIGN KEY ('ID_MAITRE') REFERENCES 'base'.'MAITRE'
('ID_MAITRE') ON DELETE CASCADE
);
```

8. Traduction d'une association 1-N



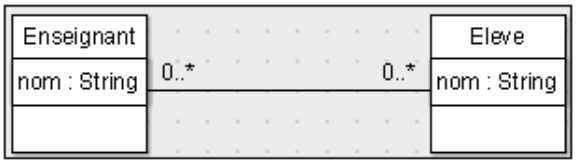
a. Traduction de classe Philosophe en code SQL

```
CREATE TABLE 'base'.'PHILOSOPHE' (
'ID_PHILOSOPHE' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
```

b. Traduction de classe Disciple en code SQL

```
CREATE TABLE 'base'.'DISCIPLE' (
'ID_DISCIPLE' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
'ID_PHILOSOPHE' INT NOT NULL,
FOREIGN KEY ('ID_PHILOSOPHE') REFERENCES 'base'.'PHILOSOPHE'
('ID_PHILOSOPHE') ON DELETE CASCADE
);
```

9. Traduction d'une association N-N



a. Traduction de la classe Enseignant en code SQL

```
CREATE TABLE 'base'.'ENSEIGNANT' (
'ID_ENSEIGNANT' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
```

b. Traduction de la classe Etudiant en code SQL

```
CREATE TABLE 'base'.'ETUDIANT' (
'ID_ETUDIANT ' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
);
```

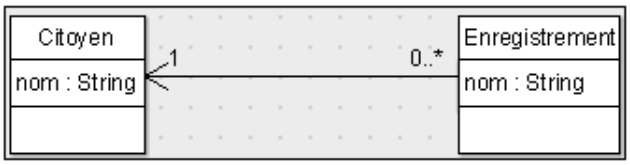
c. Traduction de l'association Etudiant/Enseignant en code SQL

```
CREATE TABLE 'base'.'ENSEIGNANT_ETUDIANT' (
'ID_ENSEIGNANT_ETUDIANT' INT NOT NULL AUTO_INCREMENT PRIMARY KEY
```


© ENI Editions - All rigths reserved - Algeria Educ

```
'ID_ENSEIGNANT' INT NOT NULL,
'ID_ETUDIANT' INT NOT NULL,

FOREIGN KEY ('ID_ENSEIGNANT') REFERENCES 'base'.'ENSEIGNANT'
('ID_ENSEIGNANT') ON DELETE CASCADE
'ID_PHILOSOPHE' INT NOT NULL,

FOREIGN KEY ('ID_ETUDIANT') REFERENCES 'base'.'ETUDIANT'
('ID_ETUDIANT') ON DELETE CASCADE
);
```

10. Traduction d'une association 1-N à navigation restreinte



Le mécanisme de traduction d'une navigation restreinte reste similaire à celui d'une navigation non restreinte. Le code ne change pas, avec ou sans navigation restreinte, pour des raisons évoquées précédemment sur le maintien de la cohérence des liens sémantiques dans un modèle relationnel classique.

11. Traduction des agrégations

Le langage SQL n'offre que très peu de moyens de traduire une distinction forte entre une association et une agrégation. Nous nous contenterons de traduire ce lien d'association fort comme nous avons déjà procédé, avec les associations par les contraintes d'intégrité et plus précisément par le dépôt de clé de référence étrangère. La seule chose que nous pouvons réaliser c'est la mise en place d'une contrainte d'intégrité spécifique indiquant que la suppression d'un tuple dans une table doit entraîner une suppression en cascade. Ce type de traduction permettra de modéliser assez finement la notion de cycle de vie lié.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction aux patrons de conception

Les patrons de conception sont des formes d'association particulière de classes permettant de faciliter l'implémentation de certaines fonctionnalités. Souvent très simples, ils ne sont pas difficiles à mettre en œuvre et facilitent grandement le travail de développement en introduisant des moyens de structurer et de manipuler aisément des objets au sein de programmes ayant des besoins particuliers.

Les patrons de conception ont les propriétés suivantes :

- Forme simple.
- Facilité d'utilisation.
- Solution à un type de problème.
- Intégration et adaptation simple dans une architecture logicielle objet.
- Combinables et flexibles.

Répondant à des objectifs distincts, chaque patron de conception est une solution possible, générique et réutilisable, pour résoudre un problème de conception logicielle. Ces patrons se décomposent en quatre grandes catégories :

- Les patrons de conception de création

Ce sont les patrons de conception permettant de gérer et de faciliter les mécanismes de création d'objets.

- Les patrons de conception de structure

Ce sont les patrons de conception permettant de gérer et de faciliter les associations entre objets.

- Les patrons de conception de comportement

Ce sont les patrons de conception permettant de gérer et de faciliter les échanges entre différents objets au sein d'une application.

- Les patrons de conception de concurrence

Ce sont les patrons de conception permettant de gérer et de faciliter les traitements dans un environnement multiprocessus.

Les patrons de conception présentés dans ce livre ont pour but de résoudre les problèmes les plus courants dans le développement en environnement orienté objet :

- Limiter le nombre d'instances d'une classe.
- Faciliter l'accès aux données.
- Création simple d'objets composites
- Faciliter la création et la manipulation d'objets.
- Structurer des chaînes de traitements.
- Mutualiser des traitements génériques.
- Homogénéiser des traitements d'objets hétérogènes.
- Simplifier la gestion de traitements complexes.

© ENI Editions - All rigths reserved - Algeria Educ

Ces exemples ne sont pas exhaustifs, il existe d'autres patrons de conception. Les patrons suivants sont des réponses à des problèmes fréquents et couvrent la plupart des besoins de développement.

© ENI Editions - All rigths reserved - Algeria Educ

Gestion de ressource limitée par Singleton

1. Le patron Singleton en quelques mots

Ce patron de conception a pour but de limiter la création d'instances d'une classe à un seul et unique objet.

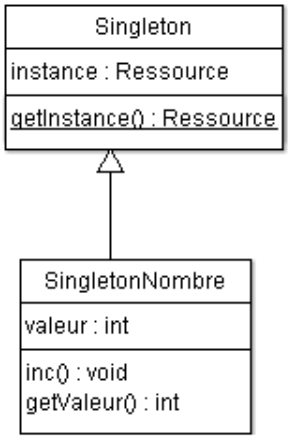
Le *Singleton* fait donc partie des patrons de conception de création car il facilite la gestion de création d'un objet unique.

Ce patron est très utile pour la gestion d'une ressource limitée telle qu'un écran d'ordinateur, un clavier, une souris ou toute autre ressource n'ayant qu'une seule et unique représentation physique ou conceptuelle.

Le principe est simple, il repose sur quelques points clés :

- Interdiction du clonage d'objet.
- Interdiction de construction d'objets par la directive new.
- Seul le premier appel a pour but de réaliser la création de l'unique instance du modèle.

a. Diagramme de classe d'un Singleton



Dans une architecture logicielle avancée, les fonctionnalités sont souvent regroupées en « services », ce qui permet une gestion centralisée des ressources et des comportements. Dans l'optique d'une organisation en service logiciel, l'utilisation du patron Singleton offre la possibilité de partager et retrouver facilement l'unique instance de chaque service de votre application. Sans ce mécanisme, de nombreuses instances des services seraient créées pénalisant les performances par surconsommation mémoire.

Le Singleton est un moyen fiable de limiter la consommation de mémoire et de limiter le temps de mise à disposition d'un objet.

---

2. Implémentation classique d'un Singleton

Cette version de la classe Singleton permet à une autre classe d'hériter des comportements classiques du Singleton.

L'exemple consiste à expliquer comment gérer une version unique d'une classe de stockage d'un entier.

La classe permet à l'objet unique d'incrémenter seulement la valeur entière et de restituer la valeur courante.

Ce type de classe est très utile pour réaliser des comptages d'instances sans avoir à stocker la valeur à l'intérieur des classes d'instances elles-mêmes.

La méthode classique du *Singleton* est *getInstance()*. Cette méthode a pour but de fournir une instance unique d'un objet. Le premier appel a en charge de créer l'unique instance de la classe et de retourner l'objet ainsi créé.

© ENI Editions - All rigths reserved - Algeria Educ

a. Classe générique d'implémentation d'un Singleton

```
<?php

class Singleton
{
     private static $instance;

     final private function __construct() { } # interdire la
surcharge du constructeur
     final private function __clone() {  } # interdire le
clonage

     final static function getInstance() {
          if (self::$instance === NULL) {
               $classe = get_called_class();
               self::$instance=new $classe();
               print "\n\t=> Création de la première
instance: $classe";
          } else {
               print "\n\t=> Récupération de l'instance
existante";
          }
          return self::$instance;
     }
}

?>
```

b. Exemple d'une classe dérivée

Par le simple mécanisme d'héritage, nous pouvons maintenant transformer une classe simple en une classe ayant le comportement type d'un Singleton.

```
<?php

class SingletonNombre extends Singleton
{
     private $val=0;

    # Méthode d'incrémentation
    public function inc() {
          $this->value++;
    }

    # Méthode de récupération de la valeur
    public function getValue() {
          return $this->value++;
     }
}
?>
```

Nous allons donc maintenant pouvoir utiliser notre classe possédant tous les avantages de notre patron Singleton.

Par héritage, tout ce qui est nécessaire est fourni et aucune autre opération ne manque.

```
<?php

# Fonction permettant le chargement au plus tard des classes
function __autoload($class_name) {
    require_once $class_name . '.class.php';
}

# Récupération de l'instance
$nb=SingletonNombre::getInstance();
$nb->inc();
$nb->inc();
```


© ENI Editions - All rigths reserved - Algeria Educ

```
print "\n * Première instance  : ". $nb->getValue();

# Récupération de l'instance
$nb2=SingletonNombre::getInstance();
$nb2->inc();
print "\n * Deuxième instance  : ". $nb2->getValue();

?>
```

c. Mise en évidence du fonctionnement du Singleton

L'exécution du programme précédent met bien en évidence l'utilisation d'une seule et même instance.

```
    => Création de la première instance: SingletonNombre
 * Première instance  : 2
     => Récupération de l'instance existante
 * Deuxième instance  : 3
```

d. Mise en évidence des restrictions par Singleton

La construction d'un objet de type SingletonNombre est prohibée :

```
# tentative de création d'une instance de Singleton
$s=new SingletonNombre();
var_dump($s);
```

```
Fatal error: Call to private Singleton::__construct() from invalid
context in .../php/patron/SingletonNombreTest.php on line 18
```

Le clonage n'entraîne pas de meilleur résultat à l'exécution.

```
# tentative de clonage du Singleton
$nb3=clone($nb2);
var_dump($nb3);
```

```
Fatal error: Call to private SingletonNombre::__clone() from
context '' in ../php/patron/SingletonNombreTest.php on line 22
```

© ENI Editions - All rigths reserved - Algeria Educ

Enrichissement des fonctionnalités par proxy

1. Le patron Proxy en quelques mots

Ce patron de conception a pour but d'enrichir le comportement d'un objet tout en conservant les interfaces qu'il expose.

Cela permet d'ajouter des comportements spécifiques sans remettre en cause l'ensemble des traitements existants ni la définition des classes existantes.

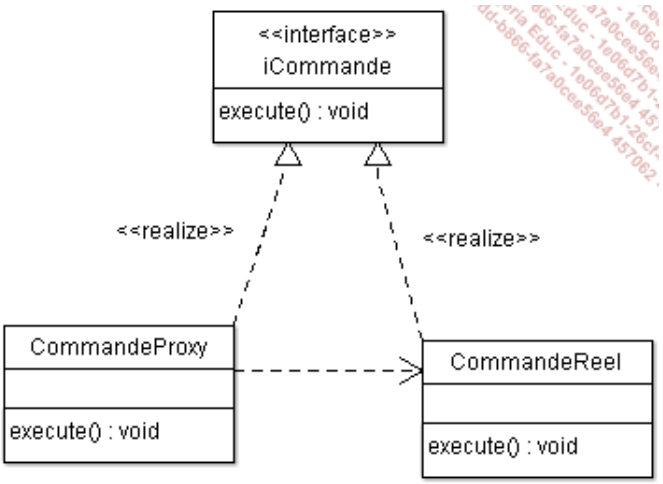
Le *Proxy* fait donc partie des patrons de conception de comportement car il facilite les relations entre les objets.

Ce patron est très utile pour la gestion uniforme de ressources hétérogènes. Il facilite la réexploitation d'interfaces de manière avantageuse tout en permettant d'enrichir le comportement d'objets existants.

Le principe est simple, il repose sur quelques points clés :

- Exposition des interfaces existantes de l'objet encapsulé.
- Enrichissement des comportements.
- Transparence de l'utilisation.

a. Diagramme de classe d'un patron Proxy



Le patron Proxy est l'outil idéal pour faciliter des transitions d'architectures en conservant la compatibilité des traitements entre deux systèmes.

2. Implémentation classique d'un proxy

Cet exemple met en évidence l'utilisation du patron Proxy dans un environnement de gestion de commandes.

Il est possible, au travers d'une interface, d'exécuter des opérations sur celle-ci.

La première version de notre système n'utilise que des objets de type CommandeReel et les évolutions d'architectures ont contraint à ajouter des traitements sur certaines commandes uniquement au travers d'une classe CommandeProxy.

a. Interface iCommande

```
<?php
```

© ENI Editions - All rigths reserved - Algeria Educ

```
interface iCommande {
        public function execute($name);
}
?>
```

b. Implémentation d'une Commande réelle

```
<?php
class CommandeReel implements iCommande
{
        final public function execute($name) {
                print "\n\t".__CLASS__." => J' execute $name \n";
        }
}
?>
```

c. Implémentation du Proxy de Commande

```
class CommandeProxy implements iCommande
{
        private $objetReel;
        final public function execute($name) {
                print "\n";
                print __CLASS__." => Avant invocation d'execute
sur \$objetReel";
                $this->objetReel->execute($name);
                print __CLASS__." => Apres invocation d'execute
sur \$objetReel \n";
        }
        public function __construct() {
                $this->objetReel=new CommandeReel();
        }
}
?>
```

d. Fonctionnement simple d'un proxy

Ce premier programme met en évidence le fonctionnement basique des classes utilisées.

```
<?php
function __autoload($class_name) {
    require_once $class_name . '.class.php';
}
# Création d'un objet CommandeReel
print "Utilisation directe de CommandeReel";
$cr=new CommandeReel();
$cr->execute("Valider");

# Création d'un objet CommandeProxy
print "\nUtilisation d'un objet CommandProxy";
$cp=new CommandeProxy();
$cp->execute('Commander');
?>
```

L'exécution montre que l'utilisation de chaque classe est identique mais que le résultat diverge entre les implémentations de *CommandeReel* et de *CommandeProxy*.

```
Utilisation directe de CommandeReel
        CommandeReel => J' execute Valider
Utilisation d'un objet CommandProxy
CommandeProxy => Avant invocation d'execute sur $objetReel
        CommandeReel => J' execute Commander
```


© ENI Editions - All rigths reserved - Algeria Educ

```
CommandeProxy => Apres invocation d'execute sur $objetReel
```

e. Collection et utilisation de Proxy

L'exemple suivant met bien en évidence qu'il est possible de gérer des objets de type *CommandeReel* et *CommandeProxy* de manière homogène en passant par le seul type commun au deux classes : *iCommande*.

Le programme se décompose en deux parties :

- Création une fois sur deux d'un objet de type *CommandeReel* ou de type *CommandeProxy*.
- Manipulation de l'ensemble des objets via la fonction *execute()* de l'interface *iCommande* de manière indépendante du type réel de l'objet.

```
<?php

function __autoload($class_name) {
    require_once $class_name . '.class.php';
}

print "\nRemplissage d'une collection de commandes";
$commandes=array();
for ($i=0; $i<6; $i++) {
     if ($i%2==0) {
          $cmd=new CommandeReel();
     } else {
          $cmd=new CommandeProxy();
     }

     print "\n* Ajout de la commande n°".($i+1);

     array_push($commandes, $cmd);
}

print "\n\n Traitement des commandes";
# Manipulation de CommandeReel et de CommandeProxy de manière
transparente
for ($i=0; $i<6; $i++) {
     print "\n* Lancement de la facturation de la commande
n°".($i+1);
     $commandes[$i]->execute("Facturer");
}
?>
```

```
Remplissage d'une collection de commandes
* Ajout de la commande n°1
* Ajout de la commande n°2
* Ajout de la commande n°3
* Ajout de la commande n°4
* Ajout de la commande n°5
* Ajout de la commande n°6

 Traitement des commandes
* Lancement de la facturation de la commande n°1
     CommandeReel => J' execute Facturer

* Lancement de la facturation de la commande n°2
CommandeProxy => Avant invocation d'execute sur $objetReel
     CommandeReel => J' execute Facturer
CommandeProxy => Apres invocation d'execute sur $objetReel

* Lancement de la facturation de la commande n°3
     CommandeReel => J' execute Facturer

* Lancement de la facturation de la commande n°4
CommandeProxy => Avant invocation d'execute sur $objetReel
```

© ENI Editions - All rigths reserved - Algeria Educ

```
     CommandeReel => J’ execute Facturer
CommandeProxy => Apres invocation d’execute sur $objetReel

* Lancement de la facturation de la commande n°5
     CommandeReel => J’ execute Facturer

* Lancement de la facturation de la commande n°6
CommandeProxy => Avant invocation d’execute sur $objetReel
     CommandeReel => J’ execute Facturer
CommandeProxy => Apres invocation d’execute sur $objetReel
```


© ENI Editions - All rigths reserved - Algeria Educ

Adaptateur d'accès aux objets

Ce patron de conception a pour but de simplifier la récupération et l'adaptation d'un objet entre plusieurs composants ou plusieurs systèmes de l'architecture logicielle.

L'Adaptateur met à disposition un ensemble de méthodes permettant de manipuler un objet au travers d'une interface simple facilitant la gestion des objets.

Le *AAO* (Adaptateur d'Accès aux Objets) fait donc partie des patrons de structure.

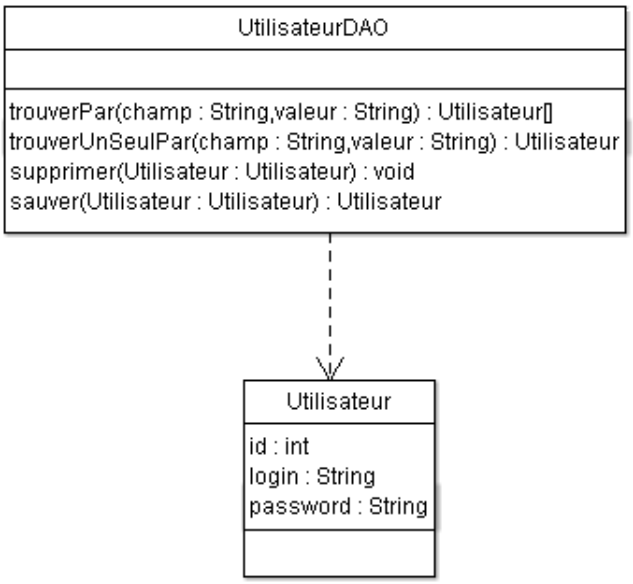
L'Adaptateur d'accès aux objets est très souvent utilisé pour gérer la persistance des objets dans une base de données. L'Adaptateur a pour but de fournir une interface simple d'ajout, de récupération, de modification et de suppression des objets pour lesquelles il réalise l'adaptation.

Son rôle consiste à manipuler les objets en fonction des données en base de données et donc d'effectuer les requêtes SQL correspondantes et de modifier le contenu des tables et des objets en fonction des appels des utilisateurs de l'Adaptateur.

Le principe est simple, il repose sur quelques points clés :

- Exposition de méthodes adaptées de manipulation de l'objet encapsulé.
- Gestion de la traduction entre le format SQL et les objets prêts à l'emploi.
- Encapsulation des fonctions de conversion et gestion de la cohérence.

1. Diagramme de classe d'un patron Adaptateur



Le patron Adaptateur est l'outil idéal pour faciliter la gestion de la persistance des objets dans une base de données. L'objet Adaptateur concentre toute la logique de conversion objet/relationnel.

2. Méthodes «standard» d'un Adaptateur

Nom de la méthode	Rôle
supprimer($objet)	*Supprimer()* recherche l'entrée et supprime les données associées.
	Sauver() ajoute les données ou modifie les données

© ENI Editions - All rigths reserved - Algeria Educ

sauver($objet)	correspondantes. Il s'agit de la méthode la plus complexe car elle vérifie la présence des entrées avant leur modification.
trouverUnSeulPar('champ', 'valeur')	*trouverUnSeulPar()* renvoie le premier élément trouvé par la méthode trouverPar().
trouverPar('champ', 'valeur')	*trouverPar()* recherche toutes les entrées en base de données correspondant à la valeur pour le champ. Cette méthode renvoie un tableau d'objets initialisés.

3. Implémentation classique d'un Adaptateur d'accès aux objets

Cet exemple met en évidence l'utilisation du patron Adaptateur dans un environnement de gestion d'utilisateurs.

La première version de notre système n'utilise que des objets de type *Utilisateur*. Il est possible d'y ajouter la manipulation de plusieurs entités ou d'y ajouter des méthodes spécifiques de recherche et de modification de données.

Il est cependant important de comprendre qu'un patron AAO n'est pas censé contenir du code « métier » ou du code de traitement des objets.

a. Schéma SQL d'Utilisateur

```
CREATE TABLE IF NOT EXISTS 'utilisateur' (
  'ID' int(11) NOT NULL AUTO_INCREMENT,
  'LOGIN' varchar(255) NOT NULL,
  'PASSWORD' varchar(255) NOT NULL,
  PRIMARY KEY ('ID'),
  UNIQUE KEY 'LOGIN' ('LOGIN')
) ;
```

b. Classe Utilisateur

```
<?php
class Utilisateur{
    private $id ;
    private $login ;
    private $password ;

    function __construct($l=NULL, $p=NULL){
        $this->id = 0 ;
        $this->login= $l ;
        $this->password = $p ;
    }

    function __set($name,$value){
        $this->$name = $value ;
    }

    function __get($name){
        return $this->$name;
    }
}
?>
```

c. Implémentation du patron AAO

© ENI Editions - All rigths reserved - Algeria Educ

```
<?php

class UtilisateurAAO{
        function __construct(){
        $this->connect = mysql_connect("localhost","dao","dao")
or die("Echec de connexion au serveur.");
        mysql_select_db("dao") or die("Echec de sélection de la
base.");
    }

    function supprimer($util){
        $requete = "delete FROM UTILISATEUR where ID=$util->id";
        mysql_query($requete,$this->connect);
    }

    function sauver($util){
        if($util->id == 0){
            $requete = "insert into UTILISATEUR
values('','$util->login','$util->password')" ;
            mysql_query($requete,$this->connect);

            $util->id = mysql_insert_id($this->connect);
        }else{
                $requete = "update UTILISATEUR set
LOGIN='$util->login', PASSWORD='$util->password' where ID=$util-
>id" ;
                mysql_query($requete,$this->connect);
        }
    return $util;
    }

    function trouverPar($champ, $val, $max=-1){
        $res=array();
        $requete = "select * from UTILISATEUR where
".strtoupper($champ)." LIKE '%$val%'" ;
        if($result = mysql_query($requete,$this->connect)) {
           while($ligne = mysql_fetch_row($result)) {
              $util = new Utilisateur();
              $util->id = $ligne[0];
              $util->login = $ligne[1];
              $util->password = $ligne[2];
              array_push($res, $util) ;
              $max--;
              if ($max==0) break;
           }
        }
        return $res ;
    }
    function trouverUnSeulPar($champ, $val){
        $res=$this->trouverPar($champ, $val, 1);
        return $res[0];
    }
}
?>
```

d. Fonctionnement de notre AAO maison

Ce premier programme de test permet de mettre en évidence le fonctionnement basique de l'Adaptateur, c'est-à-dire la sauvegarde d'un objet de type *Utilisateur*.

```
<?php

function __autoload($class_name) {
    require_once $class_name . '.class.php';
}

# Création d'un objet utilisateur
```

© ENI Editions - All rigths reserved - Algeria Educ

```
$util = new Utilisateur('Bubu', 'azerty45') ;
print "OBJET INITIAL:\n";
print_r($util) ;

$utilDAO = new UtilisateurDAO();
print "\nSauvegarde de l'utilisateur";
$util=$utilDAO->sauver($util);
print "OBJET après sauvegarde:\n";
print_r($util) ;

# Modification du mot de passe
$util->password='rienafaire';

# 2eme sauvegarde
$util=$utilDAO->sauver($util);
print "OBJET apres 2eme sauvegarde:\n";
print_r($util) ;
?>
```

L'exécution montre que la sauvegarde est différente, selon l'ajout ou la modification.

```
OBJET INITIAL:
Utilisateur Object
(
    [id:Utilisateur:private] => 0
    [login:Utilisateur:private] => Bubu
    [password:Utilisateur:private] => azerty45
)

Sauvegarde de l'utilisateur
SQL: insert into UTILISATEUR values('','Bubu5','azerty45')
OBJET après sauvegarde:
Utilisateur Object
(
    [id:Utilisateur:private] => 9
    [login:Utilisateur:private] => Bubu5
    [password:Utilisateur:private] => azerty45
)


SQL: update UTILISATEUR set LOGIN='Bubu5', PASSWORD='rienafaire'
where ID=9
OBJET apres 2eme sauvagarde:
Utilisateur Object
(
    [id:Utilisateur:private] => 9
    [login:Utilisateur:private] => Bubu5
    [password:Utilisateur:private] => rienafaire
```

e. Mise en évidence des fonctions de recherche

L'exemple suivant met en évidence qu'il est possible de rechercher simplement et efficacement des objets dans une base de données.

Le programme se décompose en trois parties :

- Une partie de recherche et de manipulation.
- Une suppression de l'objet trouvé en base.
- Une nouvelle recherche de validation.

```
<?php

function __autoload($class_name) {
    require_once $class_name . '.class.php';
```


© ENI Editions - All rigths reserved - Algeria Educ

```
}

$utilDAO = new UtilisateurDAO();

print "Recherche des logins contenant la sous-chaîne Bubu";
$resultats=$utilDAO->trouverPar('login', 'Bubu');
print_r($resultats);

print "Recherche de l'utilisateur avec le password rienafaire";
$util2=$utilDAO->trouverUnSeulPar('password', 'rienafaire');
print_r($util2);

print "Suppression de cet utilisateur";
$utilDAO->supprimer($util2);

print "Recherche de tous les utilisateurs avec le password rienafaire";
$res=$utilDAO->trouverPar('password', 'rienafaire');
print_r($res);
?>
```

```
Recherche des logins contenant la sous-chaîne Bubu
SQL: select * from UTILISATEUR where LOGIN LIKE '%Bubu%'
Array
(
    [0] => Utilisateur Object
        (
            [id:Utilisateur:private] => 3
            [login:Utilisateur:private] => Bubu
            [password:Utilisateur:private] => azerty45
        )

    [1] => Utilisateur Object
        (
            [id:Utilisateur:private] => 4
            [login:Utilisateur:private] => Bubu2
            [password:Utilisateur:private] => azerty45
        )

    [2] => Utilisateur Object
        (
            [id:Utilisateur:private] => 5
            [login:Utilisateur:private] => Bubu3
            [password:Utilisateur:private] => rienafaire
        )

)
Recherche de l'utilisateur avec le password rienafaire
SQL: select * from UTILISATEUR where PASSWORD LIKE '%rienafaire%'
Utilisateur Object
(
    [id:Utilisateur:private] => 9
    [login:Utilisateur:private] => Bubu5
    [password:Utilisateur:private] => rienafaire
)
Suppression de cet utilisateur
SQL: delete FROM UTILISATEUR where ID=9
Recherche de tous les utilisateurs avec le password rienafaire
SQL: select * from UTILISATEUR where PASSWORD LIKE '%rienafaire%'
Array
(
)
```

© ENI Editions - All rigths reserved - Algeria Educ

Gestion de la complexité par des objets composites

Ce patron de conception a pour but de simplifier la gestion des objets composites.

Le patron de conception Composite permet à un objet conteneur et à un groupe d'objets contenus d'être traités de manière similaire.

Le patron Composite réalise l'implémentation du principe ensemble/partie, tout en garantissant l'équité de traitement entre ensemble et partie.

Le Composite fait donc partie des patrons de structure.

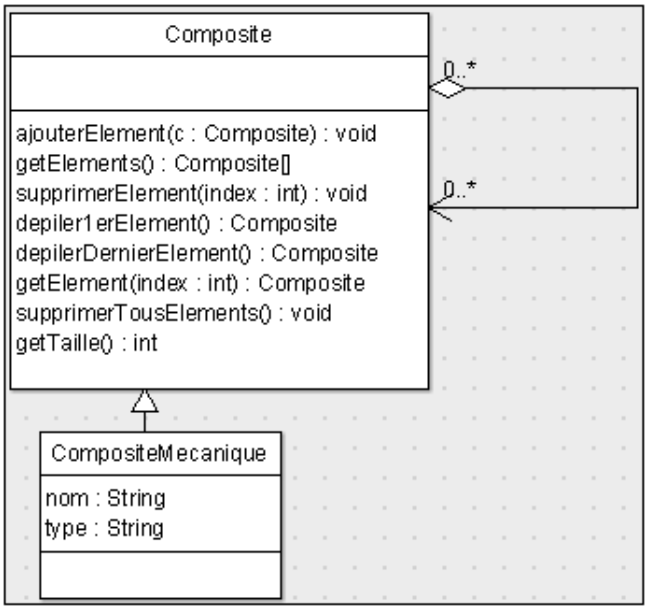
Un objet Composite est un objet ayant des caractéristiques particulières et ayant la possibilité de contenir d'autres objets composites. Ce patron est donc idéal pour créer des structures d'objets complexes sous forme d'arbres d'objets. L'objet contenant joue le rôle de nœud parent et chaque objet composite contenu joue le rôle de nœud fils.

Avec le patron Composite, l'implémentation d'un arbre n-aire devient simple.

Le principe est simple, il repose sur quelques points clés :

- Héritage d'une classe de stockage des éléments fils.
- Gestion des objets fils simplifiée.
- Encapsulation des fonctions de conversion et gestion de la cohérence.

1. Diagramme de classe d'un patron Composite



Le patron Composite est l'outil idéal pour faciliter la gestion de collection d'objets sous forme d'arbres à n fils. L'objet Composite concentre toute la logique de gestion ensemble/partie d'un objet composé d'éléments hiérarchisés.

2. Méthodes « standard » d'un objet Composite

Nom de la méthode	Rôle
getElement($index)	Cette méthode retourne l'élément à l'index $index du tableau des éléments contenus.

© ENI Editions - All rigths reserved - Algeria Educ

getElements()	Cette méthode renvoie l'ensemble des éléments contenus par l'objet.
depiler1erElement()	Cette méthode retire le premier élément contenu et le renvoie en réponse.
depilerDernierElement()	Cette méthode retire le dernier élément contenu et le renvoie en réponse.
getTaille()	Cette méthode renvoie le nombre d'éléments contenus.
supprimerTousElements()	Cette méthode supprime l'ensemble des éléments contenus.
supprimerElement($index)	Cette méthode supprime l'élément contenu à l'index $index.
ajouterElement($elt)	Cette méthode permet l'ajout d'un élément dans l'ensemble des éléments contenus.

3. Implémentation classique d'un objet Composite

Cet exemple met en évidence l'utilisation du patron Composite dans le contexte de gestion de pièces mécaniques pouvant être simples ou composées.

La version actuelle permet de créer des pièces mécaniques par assemblage.

Pour construire des objets complexes, nous associerons plusieurs objets du type CompositeMecanique qui hériteront de l'ensemble des propriétés génériques de la classe Composite.

Le patron Composite n'a pas vocation à contenir des traitements complexes. L'ensemble des traitements doivent être relégués dans une autre classe.

a. Implémentation générique d'une classe Composite

```
<?php

class Composite
{
    private $elements;

    public function __construct($init=array()) {
        $this->elements=$init;
    }
    final function getElement($i) {
        return $this->elements[$i];
    }
    final function getElements() {
        return $this->elements;
    }

    final function depiler1erElement() {
        return array_shift($this->elements);
    }
    final function depilerDernierElement() {
        return array_pop($this->elements);
    }
    final function getTaille($i) {
        return count($this->elements);
    }

    final function supprimerTousElements() {
        unset($this->elements);
        $this->elements=array();
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    }

    final function supprimerElement($elt) {
        # Cas d'un passage par entier représentant
l'index
        if (is_integer($elt)) {
            unset($this->elements[$elt]);
        # Cas d'un passage par objet
        } else { # Supprimons tous les elements
similaires
            while (true)  {
                $i = array_search( $elt, $this-
>elements );
                if ($i != NULL || $i !== FALSE) {
                    unset($this->elements[$i]);
                } else {
                    break;
                }
            }
        }
    }
    final function ajouterElement($cp_elt) {
        array_push ($this->elements, $cp_elt);
    }
}
?>
```

b. Utilisation par héritage de la classe Composite

```
<?php

class CompositeMecanique extends Composite
{
    private $type;
    private $nom;
    public function __construct($type, $nom, $elts=array()) {
        $this->nom=$nom;
        $this->type=$type;
        Composite::__construct($elts);
    }

    public function getType(){
        return $this->type;
    }

    public function getNom(){
        return $this->nom;
    }

    public function toString($esp=0) {
        for ($i=0;$i<$esp;$i++) $ret.="\t";
        $ret.=__CLASS__ . "( " . $this->type . " : " . $this-
>nom . " )";
        $ret.="\n";
        $elts=$this->getElements();
        foreach ($elts as $elt) {
            $ret.=$elt->toString($esp+1);
        }
        return $ret;
    }
}
?>
```

c. Fonctionnement de notre objet CompositeMecanique

© ENI Editions - All rigths reserved - Algeria Educ

Ce premier programme met en évidence le fonctionnement basique d'un ensemble d'objets composites, c'est-à-dire, dans cet exemple, de l'affichage de CompositeMécanique. Il met en évidence la manipulation récursive de nos objets composites.

```
<?php

function __autoload($class_name) {
    require_once $class_name . '.class.php';
}

$moteur=new CompositeMecanique('Moteur', 'Bloc');
$piston1=new CompositeMecanique('Moteur', 'Piston1');
$piston2=new CompositeMecanique('Moteur', 'Piston2');

$demarreur=new CompositeMecanique('Moteur', 'Démarreur');
$fussible=new CompositeMecanique('Electronique', 'Fussible 1A');
$fussible2=new CompositeMecanique('Electronique', 'Fussible 5A');

$demarreur->ajouterElement($fussible);
$demarreur->ajouterElement($fussible2);
$moteur->ajouterElement($demarreur);
$moteur->ajouterElement($piston1);
$moteur->ajouterElement($piston2);


print $moteur->toString();

?>
```

L'exécution montre l'affichage récursif des éléments et des parties composant notre moteur.

```
CompositeMecanique( Moteur : Bloc )
     CompositeMecanique( Moteur : Démarreur )
          CompositeMecanique( Electronique : Fussible 1A )
          CompositeMecanique( Electronique : Fussible 5A )
     CompositeMecanique( Moteur : Piston1 )
     CompositeMecanique( Moteur : Piston2 )
```


© ENI Editions - All rigths reserved - Algeria Educ

Simulation de n'importe quelle classe par imitateur

Un objet 'imitateur' ou 'moqueur' provient de l'anglais « Mock objet ». Cet objet a la capacité de simuler ou singer le comportement d'un objet n'existant pas encore dans votre système.

En effet, il est courant que plusieurs équipes se partagent les tâches de réalisation. Pendant qu'une équipe réalise l'ensemble des classes d'accès aux bases de données et la gestion de la persistance de chaque objet, une autre équipe est en charge de l'interface graphique et une autre des traitements spécifiques des données pour le client.

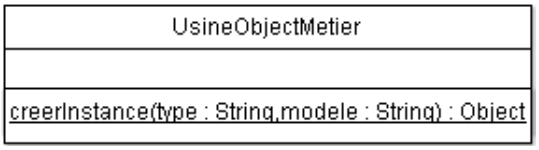
L'objet imitateur a pour travail d'exposer et de stocker toute information de manière provisoire et prévisible sans implémenter aucun mécanisme structuré de stockage d'information.

Il peut offrir des fonctionnalités importantes telles que la persistance des données et le stockage de l'état des variables.

Cependant, l'objet imitateur ne fera que simuler un contexte et des réponses stéréotypées et prédéfinies et, de ce fait, devra rapidement faire place à des vrais composants logiciels répondant pleinement à leurs fonctions.

De ce fait, il est important de prévoir une mise en place du patron de conception 'Usine' permettant de se brancher rapidement sur les composants finaux de votre architecture ou sur les composants d'imitation.

1. Le patron de conception Usine



Notre modèle d'usine est simplifié et ne possède pas de classe abstraite générique. Notre modèle s'appuie simplement sur une méthode statique ayant pour but de créer pour vous un objet particulier en effectuant l'ensemble des opérations afin qu'il soit utilisable. Dans la forme plus classique et moins raccourcie de l'Usine, celle-ci possède une classe générique UsineAbstraite qui est ensuite héritée par une classe UsineConcrète. La classe UsineConcrète devant implémenter la méthode abstraite creerInstance() de la classe mère UsineAbstraite.

Ce modèle plus léger nous permet de bâtir encore plus rapidement et simplement une Usine en PHP et à moindre coût.

En effet, la méthode statique creerInstance() comporte deux paramètres :

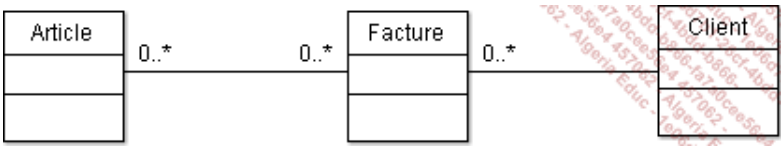
Le premier paramètre définit le type de code que l'on souhaite utiliser dans notre exemple, nous avons trois possibilités :

- Vide
- Imitateur
- Pdo

Le second paramètre concerne la classe de l'objet que l'on souhaite créer, et là encore, nous avons trois choix :

- Article
- Facture
- Client

Le modèle général décrit lors de la phase d'analyse et de conception nous décrit un modèle structuré selon ce très simple diagramme UML :



© ENI Editions - All rigths reserved - Algeria Educ

Ce modèle ne présente que les propositions suivantes :

- Un client peut avoir plusieurs factures.
- Une facture ne correspond qu'à un seul client.
- Un article peut apparaître dans plusieurs factures.
- Une facture peut contenir plusieurs articles.

2. Code de l'Usine

Le code de l'Usine est simple et doit le rester :

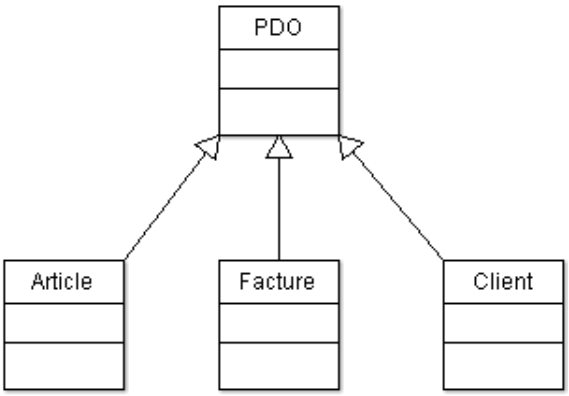
```
<?php
class UsineObjetMetier {
    public static function creerInstance($type, $modele) {
        if ( !file_exists("$modele/$type.class.php") ) {
            echo "\nWarn : type($type) manquant pour le
modèle($modele)";
            return null;
        }

        if ($modele=="imitateur") require_once (
"$modele/ImitateurGenerique.class.php" );
        if ($modele=="vide") require_once (
"$modele/ImitateurVide.class.php" );
        require_once ("$modele/$type.class.php");
        return new $type();
    }
}
?>
```

3. Utilisation de l'Usine

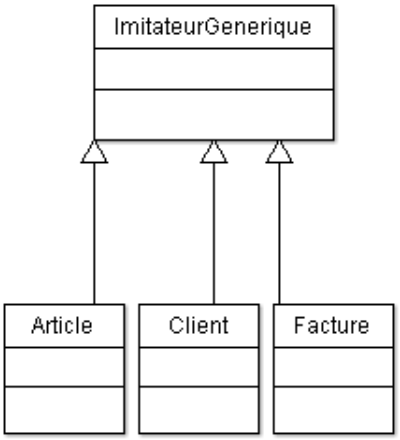
Selon la valeur de la variable $modele, les classes instanciées par l'Usine ne seront pas les mêmes.

Nous pourrons donc faire coexister au moins deux modèles d'implémentation de vos classes sans que cela ne pose aucun souci d'intégration de nouvelles implémentations.

Ainsi les deux modèles suivants, quoique antagonistes, peuvent exister dans un même projet.



© ENI Editions - All rigths reserved - Algeria Educ




```
<?php
require_once ( "./UsineObjetMetier.class.php" );

$modele="imitateur";
#$modele="vide";
#$modele="pdo";

echo "<pre>";
$a1=UsineObjetMetier::creerInstance("Article", $modele);
$a1->setLabel("Pot de Confitures");
$a1->setPrixHt(12.4);


$a2=UsineObjetMetier::creerInstance("Article", $modele);
$a2->setLabel("6 Bananes");
$a2->setPrixHt(3.0);

$a3=UsineObjetMetier::creerInstance("Article", $modele);
$a3->setLabel("250G beurre");
$a3->setPrixHt(2.0);

$ft=UsineObjetMetier::creerInstance("Facture", $modele);
$ft->setId(1);
$ft->setDate(date("D M j G:i:s T Y"));
$ft->addProduits($a1);
$ft->addProduits($a3);

$ft2=UsineObjetMetier::creerInstance("Facture", $modele);
$ft2->setId(2);
$ft2->setDate(date("D M j G:i:s T Y"));
$ft2->addProduits($a2);

$ft3=UsineObjetMetier::creerInstance("Facture", $modele);
$ft3->setId(3);
$ft3->setDate(date("D M j G:i:s T Y"));
$ft3->addProduits($a1);
$ft3->addProduits($a3);
$ft3->addProduits($a2);

$cl=UsineObjetMetier::creerInstance("Client", $modele);
$cl->setNom("Renouard");
$cl->setPrenom("J.Marie");
$cl->setSiret("123456789");
$cl->addFactures($ft);
$cl->addFactures($ft2);
$cl->addFactures($ft3);
echo "\n";
echo $cl;

Client::save("c1", $cl);

$cl2=Client::load("c1");
```

© ENI Editions - All rigths reserved - Algeria Educ

```
echo "\n$cl2";
if (!isset($cl2)) exit;
foreach ($cl2->getFactures() as $f) {
    print "<li>$f";
        foreach ($f->getProduits() as $pdt) echo "\n\n\t\t$pdt";
    echo "</li>";
}
echo "</pre>";
exit(0);
?>
```

4. Résultat du programme de test

```
· Facture Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Id] => 1
            [Date] => Sat Jan 30 0:33:31 UTC 2010
            [Produits] => Array
                (
                    [0] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => Pot de Confiture
                                    [PrixHt] => 12.4
                                )
                        )

                    [1] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => 250G beurre
                                    [PrixHt] => 2
                                )
                        )
                )
        )
)
        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => Pot de Confiture
            [PrixHt] => 12.4
        )

)
        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => 250G beurre
            [PrixHt] => 2
        )
)
·  Facture Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Id] => 2
            [Date] => Sat Jan 30 0:33:31 UTC 2010
```


© ENI Editions - All rigths reserved - Algeria Educ

```
            [Produits] => Array
                (
                    [0] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => 6 Bananes
                                    [PrixHt] => 3
                                )
                        )
                )
        )
)
        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => 6 Bananes
            [PrixHt] => 3
        )
)
·  Facture Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Id] => 3
            [Date] => Sat Jan 30 0:33:31 UTC 2010
            [Produits] => Array
                (
                    [0] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => Pot de Confiture
                                    [PrixHt] => 12.4
                                )
                        )
                    [1] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => 250G beurre
                                    [PrixHt] => 2
                                )
                        )

                    [2] => Article Object
                        (
                            [data:ImitateurGenerique:private] =>
Array
                                (
                                    [Label] => 6 Bananes
                                    [PrixHt] => 3
                                )
                       )
                )
        )
)

        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => Pot de Confiture
            [PrixHt] => 12.4
        )
```

© ENI Editions - All rigths reserved - Algeria Educ

```
)

        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => 250G beurre
            [PrixHt] => 2
        )
)

        Article Object
(
    [data:ImitateurGenerique:private] => Array
        (
            [Label] => 6 Bananes
            [PrixHt] => 3
        )
)
```

5. Code d’un imitateur vide

```
<?php

class ImitateurVide {

    public function __call($f, $params) {
        return array();
    }

    public function __toString() {
        return "";
    }
    public static function save($f, $o) {
    }

    public static function load($f) {
    }
}
?>
```

6. Code de la classe imitateur générique

```
<?php

class ImitateurGenerique {
    private $data;
    public function __construct($initValues=null) {
        $this->data=array();
        if ($initValues!=null and is_array($initValues) ) {
            foreach ($initValues as $v => $k)
                $this->data[$v]=$k;
        }
    }

    public function __get($name) {
        echo "ERREUR: accès à $name interdit ( concept
d’encapsulation )";
        exit(0);
    }
    public function __set($name, $value) {
        echo "ERREUR: accès à $name interdit ( concept
d’encapsulation ) valeur : $value non prise en compte ";
```


© ENI Editions - All rigths reserved - Algeria Educ

```
        exit(0);
    }
    public function __toString() {
        return print_r($this, TRUE);
    }

    public static function save($cle, $obj) {
        $filename="C:\\temp\\$cle.obj";
        if (file_exists($filename)) unlink($filename);
        file_put_contents($filename, serialize($obj));
    }

    public static function load($cle) {
        $filename="C:\\temp\\$cle.obj";
        if (!file_exists($filename)) return null;
        return unserialize(file_get_contents($filename));
    }

    public function __call($f, $params) {
        $label=substr($f,0,3); //les 3 premières lettres du nom
de la méthode
        $vari=substr($f,3); // les autres lettres

        switch ($label) {
            case "set":
                $this->data[$vari]=$params[0];
                break;

            case "get":
                if (isset($this->data[$vari]))
                    return $this->data[$vari];
                return null;
                break;
            case "add":
                // Premier passage création du tableau php
                if (!isset($this->data[$vari]))
$this->data[$vari]=array();

                //Ajout uniquement d'éléments uniques
                foreach ($params as $p) {
                    if (!in_array($p, $this->data[$vari]))
                        array_push($this->data[$vari],
$p);
                    else
                        echo "\nWarn: $p already in
$vari";
                }
                break;
            case "del":
                if ( isset ($this->data[$vari]) and
is_array($this->data[$vari]) )
                $this->data[substr($f,3)]=array_diff($this-
>data[$vari], $params);
                break;
            default:
                // rien n'est fait dans les autres cas
        }
    }
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

Création d'un pipeline de commande

Le pipeline de commande s'appuie en fait sur deux patrons distincts :

Le premier est le patron nommé Commande permettant de créer facilement des commandes génériques et le deuxième est le patron chaîne de responsabilité permettant de chaîner l'ensemble des commandes.

Le premier principe est que chaque tâche possède la même forme dans l'ensemble de l'application. Cela simplifie grandement la cohérence du code et la maintenance. En effet, il est largement plus simple de tester unitairement des commandes ayant toutes la même forme que d'écrire des tests unitaires pour du code ayant des interfaces hétérogènes. De plus, chaque ajout ou intégration de code externe peut faire l'objet d'une intégration rapide au sein de l'architecture logicielle en l'emballant dans le code d'une commande.

La souplesse viendra automatiquement avec l'application de ce patron car, en fait, plus rien ne pourra faire adhérence dans votre application. En effet, si votre architecture logicielle vous permet de changer une partie de votre logiciel en modifiant un paramètre tout en garantissant que l'ensemble reste cohérent et fonctionne correctement, alors vous bénéficierez des avantages du découpage logiciel. Ce découpage vous permettra d'intégrer rapidement de nouvelles briques logicielles plus performantes et surtout de produire des tests unitaires standard indépendants de vos composants.

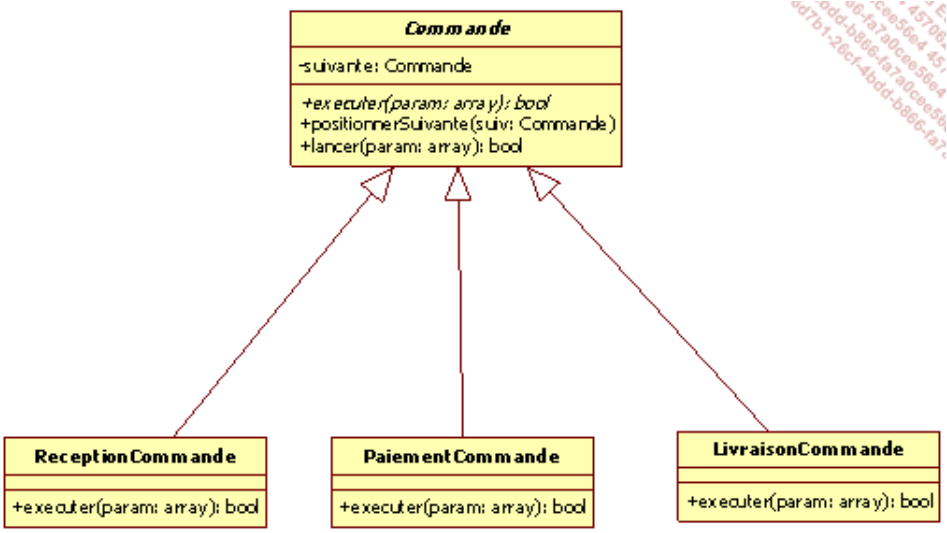
Le but du patron issu de la fusion de ces deux patrons est de créer un chaînage d'opérations ayant une forme identique (Commande) et de pouvoir finalement rapidement configurer notre chaîne via un fichier de configuration.

Le patron Commande apporte la forme classique de la méthode executer() commune à chaque commande et le patron chaîne de responsabilité apporte les méthodes positionnerSuivante() et lancer().

La forme finale du patron chaîne de commande repose sur :

- Une forme unique pour l'utilisation.
- La connaissance de l'étape suivante.

1. Le patron de conception chaîne de commande modifié



Dans notre conception, nous avons choisi de faire porter la forme des méthodes par une classe abstraite mère Commande.

La méthode lancer() de Commande a pour but de s'exécuter elle-même et, une fois terminée, de chercher la commande suivante et de lancer son exécution.

La méthode executer() a pour but de réaliser un traitement sur un tableau de paramètres. Ce tableau de paramètres est souvent appelé contexte d'exécution.

Pour tester une commande de manière unitaire, il suffit donc de créer un contexte, sous forme de tableau PHP, et d'invoquer la méthode executer. Puis de vérifier deux points importants :

- Le retour de la méthode executer() à true.

© ENI Editions - All rigths reserved - Algeria Educ

- Le changement des informations dans le contexte.

2. Code de la classe mère Commande

```
<?php

abstract class Commande {
    private $commandeSuivante;

    public abstract function executer( array &$contexte ) ;

    protected function debug($info) {
        echo "\n* ".date("j-m-Y_h:i:s")." ".get_called_class().
" ".$info;
    }
    public function lancer( array &$contexte=array(),
$debug=false)
    {
        if ($debug) $this->debug("DEBUG ACTIVE");

        //stocker le contexte avant l'exécution
        $c_avant=print_r($contexte, true);
        if ($debug) $this->debug("Contexte AV: ".$c_avant);
        //Execution de l'exécution de la commande
        $result=$this->executer($contexte);

        //Si l'exécution n'a pas fonctionné
        if ( $result == FALSE ) {
            //Stocker contexte apres erreur
            $c_apres=print_r($contexte, true);

            // Positionnement des informations d'erreur
            $contexte["CMD_ERREUR"]=get_called_class();
            $contexte["CONTEXTE_ERREUR_AVANT"]=$c_avant;
            $contexte["CONTEXTE_ERREUR_APRES"]=$c_apres;

            //Renvoyer FALSE
            return FALSE;
        }
        $c_avant="";
        if ($debug) $this->debug("Commande OK");
        //S'il y a une commande suivante
        if ( isset($this->commandeSuivante) ) {
                //lancer la commande suivante
                if ($debug) $this->debug("Contexte
Lancement commande suivante: ".print_r($this->commandeSuivante,
true));

                $result=$this->commandeSuivante-
>lancer($contexte, $debug);
        }

        return $result;
    }

    public function positionnerSuivante(Commande $suivante) {
        $this->commandeSuivante=$suivante;
    }
}
?>
```

3. Code de la classe LivraisonCommande

```
<?php
```


© ENI Editions - All rigths reserved - Algeria Educ

```
class LivraisonCommande extends Commande {

    public function executer( array &$contexte ) {
        if (!isset($contexte["NbPiece"])) {
            $this->debug("Contexte incomplet: NbPiece
manquant");
            return FALSE;
        }
        $contexte["Prix"]=2*$contexte["NbPiece"];
        $this->debug( "\n################################");
        $this->debug( "\n\t".__CLASS__.": Livraison Commande");
        $this->debug( "\n################################\n");
        $this->debug(print_r($contexte, true));
        $this->debug( "\n################################");

        if (!isset($contexte["Prix"])) {
            $this->debug("Contexte incomplet en sortie : Prix
manquant");
            return FALSE;
        }
        return TRUE;
    }
}
?>
```

4. Code de la classe ReceptionCommande

```
<?php

class ReceptionCommande extends Commande {

    public function executer( array &$contexte ) {
        echo "\n################################";
        echo "\n\t".__CLASS__.": Reception Commande";
        echo "\n################################\n";
        print_r($contexte);
        echo "\n################################";
        return TRUE;
    }
}
?>
```

5. Code de la classe ErreurAleatoireCommande

```
<?php

class ErreurAleatoireCommande extends Commande {

    public function executer( array &$contexte ) {
        $res=rand(0,1);
        $this->debug("Renvoyer: $res");
        if ($res==0) {
            $contexte["NbPiece"]=-1;
            return FALSE;
        }
        return TRUE;
    }
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

6. Exemple de fichier de configuration

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Commandes>
    <Commande id="30">
        <nom>Commande de livraison</nom>
        <classe>LivraisonCommande</classe>
    </Commande>
    <Commande id="10">
        <nom>Commande de reception</nom>
        <classe>ReceptionCommande</classe>
    </Commande>
    <Commande id="50">
        <nom>Commande de livraison 2</nom>
        <classe>LivraisonCommande</classe>
    </Commande>
    <Commande id="20">
        <nom>Commande de erreur aléatoire</nom>
        <classe>ErreurAleatoireCommande</classe>
    </Commande>
    <Commande id="40">
        <nom>Commande de livraison 2</nom>
        <classe>LivraisonCommande</classe>
    </Commande>

</Commandes>
```

7. Code de la classe UsineChaine

```
<?php

class UsineCommande {
    public static function creerInstance($config, $debug=false) {
        if ( !file_exists("$config") ) {
            echo "\nWarn : config $config manquant ";
            return null;
        }
        $xml = simplexml_load_file($config);


        $tabClasse=array();
        foreach ($xml->Commande as $cmd) {
            $attrs=$cmd->attributes();
            $id=$attrs['id'];
            if ($debug) echo "\n* Création de la commande:
".$cmd->nom. " ($id)";
            $class_name=$cmd->classe->__toString();
            $objCmd= new $class_name();
            $tabClasse["$id"]=$objCmd;
        }
        krsort($tabClasse);
        if ($debug) print_r($tabClasse);
        $prev=NULL;
        $last=NULL;
        foreach ( $tabClasse as $id => $cmd) {
            if ($debug) echo "\n$id) ".print_r($cmd, true);
            if ($last !=NULL) $cmd-
>positionnerSuivante($last);
            $last=$cmd;
        }
        return $last;
    }
}
?>
```


© ENI Editions - All rigths reserved - Algeria Educ

8. Code de la classe TestPaiementCommande

```
<?php

function __autoload($class_name) {
    require_once ( "./$class_name.class.php" );
}

$chaine=UsineCommande::creerInstance("config.xml");

//print_r($premiereCommande);
$contexte=array(
                "Produit" => "Bière",
                "NbPiece" => 50
            );

$chaine->lancer($contexte);

print_r($contexte);
?>
```

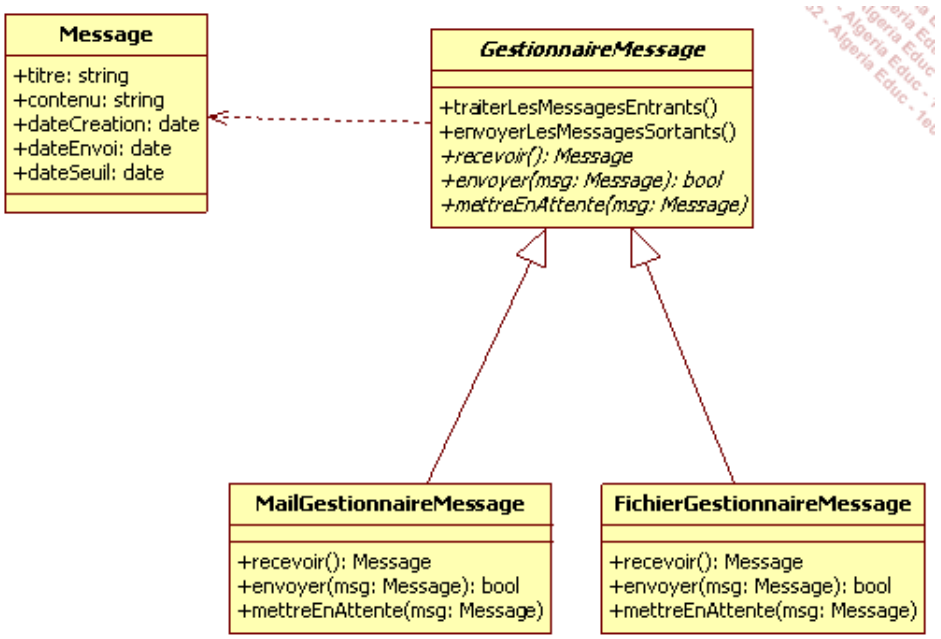
© ENI Editions - All rigths reserved - Algeria Educ

Introduction de Modèle dans votre code

Ce patron a pour but de proposer un traitement générique en laissant le soin à la classe dérivée de gérer l'implémentation des méthodes de base.

Dans notre exemple, nous avons une classe générique Gestionnaire de messages qui est capable de gérer la réception de messages. En effet, les messages qui arrivent doivent être, soit validés (c'est-à-dire avoir l'ensemble de leurs informations valides), soit rejetés. La fonction envoyer(), quant à elle, est plus complexe : les messages n'ayant pas franchi la date seuil ne doivent pas être envoyés mais mis en attente avant envoi. Les messages ayant une date de seuil vide ou ayant une date de seuil dépassée doivent être envoyés.

1. Le patron de conception Modèle



2. Code de la classe Message

```
<?php

class Message {
    private $titre;
    private $contenu;
    private $dateCreation;
    private $dateEnvoi;
    private $dateSeuil;

    public function set($var, $val) {
        $this->$var=$val;
    }

    public function get($var) {
        return $this->$var;
    }

    public function __construct($titre="", $contenu="") {
        $this->titre=$titre;
        $this->contenu=$contenu;
    }
}
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

3. Code du gestionnaire de message

```
<?php

abstract class GestionnaireMessage {
    public abstract function recevoir();
    public abstract function envoyer($msg);
    public abstract function mettreEnAttente($msg);

    protected function debug($info) {
        echo "\n* ".date("j-m-Y_h:i:s")." ".get_called_class().
" ".$info;
    }
    public function traiterLesMessagesEntrants(){
        while ( ($msg=$this->recevoir()) != NULL) {
            $status="ERREUR";
            if (
                $msg->get('titre') !=NULL &&
                $msg->get('contenu')!=NULL
            ) {
                $status="OK";
            }
            $this->debug ("Reception $status du message :
".print_r($msg, true));
        }
    }

    public function traiterLesMessagesSortants(){
        while ( ($msg=$this->recevoir()) != NULL) {
            $status="ERREUR";
            if (
                $msg->get('titre') !=NULL &&
                $msg->get('contenu')!=NULL
            ) {
            $status="OK";
            }
            $this->debug ("Reception $status du message :
".print_r($msg, true));

            if ($status=="OK"){
                if ( isset($dateSeuil) && DateTime::diff(
new DateTime("now"), $dateSeuil)< 0) {
                    $this->debug ("Mise en attente du
message: ".print_r($msg, true));
                    $this->mettreEnAttente($msg);
                } else {
                    $this->envoyer($msg);
                }
            } else {
                $this->debug ("Reception $status du message
 : ".print_r($msg, true));
            }
        }
    }
}
?>
```

4. Code du gestionnaire de message fichier

```
<?php

class FichierGestionnaireMessage extends GestionnaireMessage {
    private $msgDir;
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    public function __construct($dir="/tmp/msg") {
        if (!is_dir($dir)) mkdir($dir,0777 , true);
        $this->msgDir=$dir;
    }

    private function parseMessage($f) {
        $content=file_get_contents($f);

        $ligne=split ("\n", $content);
        $titre=$ligne[0];
        $ligne[0]="";
        $message= implode ("\n", $ligne);
        return new Message($titre, $message);
    }
    public function recevoir() {
        $msg=NULL;
        if ($handle = opendir($this->msgDir)) {
            while (false !== ($file = readdir($handle))) {
                $file=$this->msgDir."/$file";
                echo "* => $file\n";

                if (is_file($file) && strstr($file, ".msg" ) !=
FALSE ) {
                        echo "\t\t->Je traite $file\n";
                        $msg=$this->parseMessage($file);
                    print_r($msg);
                    unlink($file);
                    break;
                    }
                    }
        closedir($handle);
        }
        return $msg;
    }

    public function envoyer($msg){

    }
    public function mettreEnAttente($msg){

    }
}
?>
```

5. Exemple d’utilisation

```
<?php


function __autoload($class_name) {
    require_once ( "./$class_name.class.php" );
}

$gm=new FileGestionnaireMessage();
$gm->traiterLesMessagesEntrants();

$gm->traiterLesMessagesSortants();
?>
```

© ENI Editions - All rigths reserved - Algeria Educ

Introduction

Ce chapitre évoque le dilemme entre deux choix : intégrer des technologies externes ou bien maîtriser une application en la codant de A à Z. Nous évoquerons les avantages et les inconvénients de ces deux possibilités.

© ENI Editions - All rigths reserved - Algeria Educ

Les paradigmes de développement

Selon Wikipédia, le paradigme se définit comme suit :

« Un paradigme est une représentation du monde, une manière de voir les choses, un modèle cohérent de vision du monde qui repose sur une base définie (matrice disciplinaire, modèle théorique ou courant de pensée). C'est en quelque sorte un rail de la pensée dont les lois ne doivent pas être confondues avec un autre paradigme. »

Il existe plusieurs paradigmes de développement donc plusieurs modèles de réalisation et de mise en œuvre de solution logicielle en PHP.

- Le paradigme du tout intégré.
- Le paradigme du tout spécifique.
- Le paradigme du SAAS.
- Le paradigme hybride intégré/spécifique.

© ENI Editions - All rigths reserved - Algeria Educ

Le paradigme du tout intégré

Cette vision est une vision fortement répandue consistant à choisir un produit logiciel existant. Puis d'y réaliser un certain nombre d'adaptations, d'extensions et d'ajouts spécifiques afin de produire la solution que l'on souhaite obtenir.

Ce type de choix stratégique permet d'obtenir des résultats initiaux très intéressants et surtout très rapides. Cependant, il existe des fondamentaux qu'il ne faut jamais remettre en cause, comme par exemple, le respect des normes et des bonnes pratiques de personnalisation du logiciel.

Dans le logiciel Drupal, système de gestion de contenu en PHP, l'ensemble des modifications ainsi que l'ajout d'extensions doivent suivre des règles et une démarche spécifique à Drupal. Cette démarche est nommée « The Drupal Way », l'esprit Drupal. Ce que souhaitent promouvoir les développeurs de solutions complètes, c'est un moyen de garantir l'évolutivité des adaptations au travers de l'évolution des versions du produit.

1. Problèmes d'intégration de solutions complètes

S'il est exagéré de tout écrire et de ne rien réutiliser lors d'un projet en pensant que le travail des autres est de moins bonne qualité, il est aussi dommage de penser qu'utiliser une application existante va répondre simplement et rapidement à l'intégralité de vos besoins.

En effet, il n'y a rien de pire que de croire qu'une entreprise peut fonctionner sur un ensemble de produits PHP installés les uns à côté des autres.

Il existe de nombreux développements spécifiques permettant d'harmoniser un ensemble de solutions complètes. Ces solutions sont très sensibles car elles dépendent entièrement des versions spécifiques de votre solution. Elles nécessitent une connaissance avancée des logiciels ou provoquent une forte envie de venir plonger son esprit dans le code PHP des réalisateurs du logiciel, quand celui-ci est disponible.

Il est donc très important de valider les compétences des intervenants techniques venant mettre en place ce type de solution. En effet, rien n'est plus néfaste qu'un prestataire venant « se faire la main » sur votre plate-forme et réalisant des mises en place hasardeuses.

Un principe : validez les compétences et surtout les expériences précédentes de votre prestataire avant de le laisser mettre en place votre solution « clé en main ».

Il en résulte un certain nombre de points chauds qu'il faut identifier rapidement tels que :

- Viabilité du produit sélectionné.
- Coût et disponibilité des compétences techniques.
- Pérennité des deux indicateurs précédents.

2. Évaluation d'une solution libre

La première des étapes dans la recherche d'une solution clé en main est la recherche des produits libres (« open source ») existants et l'évaluation des potentialités de base offertes par ce type de produit. Une fois que vous connaissez exactement le potentiel d'un produit libre, si vous souhaitez investir dans un logiciel payant, vous aurez les clés pour évaluer ce qui est standard (ou par défaut) de ce qui est de l'ordre de l'innovation et de l'avance technologique.

Il y a cependant aujourd'hui suffisamment de ressources libres pour bâtir la solution PHP adaptée à votre besoin.

a. Comment évaluer ces solutions ?

La méthode proposée est assez simple. C'est une version personnelle et dérivée directement de OSMM (*Open Source Maturity Model*) qui est un modèle de maturité de solution open source.

L'idée est de mesurer objectivement et subjectivement dix indicateurs de performance sur une échelle de 0 à 10.

Ces critères, une fois évalués, peuvent être mis en relief au travers d'un graphique permettant une représentation visuelle de la pertinence d'une solution logicielle. La mise en relief peut aussi combiner plusieurs évaluations afin de mettre en avant les avantages et les défauts des solutions les unes par rapport aux autres. Le but de cette approche est de fournir une base permettant des choix les plus objectifs possibles.

© ENI Editions - All rigths reserved - Algeria Educ

b. Critères d'évaluation

Voici donc les dix clés permettant d'évaluer la pertinence du choix d'un logiciel libre :

La communauté

Le nombre de développeurs, d'utilisateurs et le nombre d'inscrits aux newsletters et aux listes de diffusion sont des critères qui permettent d'évaluer la taille de la communauté.

Le nombre de téléchargements des différentes versions du logiciel est aussi très important.

Il est utile de comparer ce nombre de téléchargements au nombre de téléchargements de logiciels similaires.

Les questions qu'il faut se poser : est-ce le logiciel le plus populaire de sa catégorie ? Est-il utilisé régulièrement pour résoudre les mêmes problématiques que celles de mon projet ?

L'activité des versions

Le nombre de versions et les planifications des versions (roadmap) sont autant d'indices mettant en avant la vélocité d'une équipe de développement. En effet, les logiciels libres ont un leitmotiv : « realease often release early », qui invite à sortir de nouvelles versions du logiciel rapidement (c'est-à-dire dès qu'elles sont disponibles).

Plus il y a de versions, plus le projet semble actif et donc plus il attire d'utilisateurs.

La longévité

L'une des questions les plus importantes est : quand ce projet est-il sorti pour la première fois ?

Linux, le système d'exploitation libre, a 19 ans. Le Langage PHP, le serveur HTTP Apache et le serveur MySQL ont tous les trois 15 ans.

La longévité n'est pas gage de qualité. Cependant, il est important d'utiliser des projets ayant un minimum d'un an de vie afin de ne pas être pris dans un effet de mode provisoire qui peut vous faire choisir un logiciel attrayant mais sans pérennité possible.

Les licences

La licence est très importante. En effet, il est intéressant que le logiciel possède une version commerciale du produit afin de garantir des revenus qui vont permettre de financer les évolutions des futures versions ainsi que l'emploi des développeurs.

La licence libre utilisée est aussi un point de détail essentiel pouvant même vous empêcher de commercialiser votre logiciel avec une version embarquée de ce logiciel. Soyez prudent et vigilant sur ce détail.

L'existence de plusieurs licences est signe de pérennité pour un logiciel libre.

Le support

Y a-t-il des forums, des FAQ, des listes de diffusion ? Quelles sont les activités sur ces supports ? Combien de messages sont déposés au travers de ces médias ?

Combien de sites extérieurs présentent et commentent le logiciel ou le moyen de le mettre en place ?

La documentation

Existe-t-il une documentation utilisateur, un manuel des développeurs, des tutoriaux sur le produit ?

Un logiciel sans documentation est simplement inutilisable car il est impossible de comprendre son fonctionnement sans ce manuel.

Il faut au minimum un manuel utilisateur permettant de comprendre comment utiliser le logiciel et un manuel d'installation et de paramétrage permettant d'adapter le logiciel à votre besoin.

Les failles de sécurité

Les failles de sécurité font-elles l'objet de traitements spécifiques leur permettant d'être corrigées rapidement sur vos installations ?

Il est dont important d'avoir cette vision de la sécurité car le logiciel que vous utilisez peut être utilisé comme tremplin vers vos bases de données et permettre de récupérer, de corrompre ou de détruire vos données par l'utilisation d'attaques spécifiques à ce logiciel.

Les fonctionnalités

© ENI Editions - All rigths reserved - Algeria Educ

Combien de fonctionnalités sont implémentées et à quelle fréquence ?

Les fonctionnalités font et défont un logiciel. Un logiciel n'apportant pas de services ou de fonctionnalités suffisantes ne peut pas prétendre être un bon candidat pour votre projet car il ne sera pas capable de supporter des fonctions annexes qui lui seront ajoutées en cours de projet.

Il est donc important qu'un projet offre un nombre de fonctions suffisantes sans que cela nuise au projet en lui-même ni à sa qualité.

L'intégration

L'utilisation d'une technologie standard et d'une architecture logicielle moderne et modulaire sont de bons signes d'intégration indiquant que le produit fonctionne bien avec d'autres produits (et sera donc susceptible de bien fonctionner avec vos autres produits). L'utilisation de protocoles standard JSON, HTTP ou XML est aussi importante et indique que le produit ne souhaite pas se lancer dans une piste d'innovation pure et qu'il a des objectifs de pérennité et de fonctionnement dans les environnements modernes de l'informatique.

L'objectif et l'origine

Ce point est important car il détermine la volonté, l'objectif et le sens de l'évolution du produit. Si votre objectif est de produire un logiciel stable dans le temps et que le projet a une vision d'innovation sans souci de la notion d'évolution, il faudra être vigilant que ce point ne devienne pas un fossé infranchissable par la suite et ne vous pousse pas à maintenir une version obsolète du logiciel.

c. Script de génération d'un graphe

Afin de permettre la visualisation de votre étude et de mettre en relief votre analyse, voici un script capable de prendre en compte vos notes et de produire un graphe de visualisation des résultats.

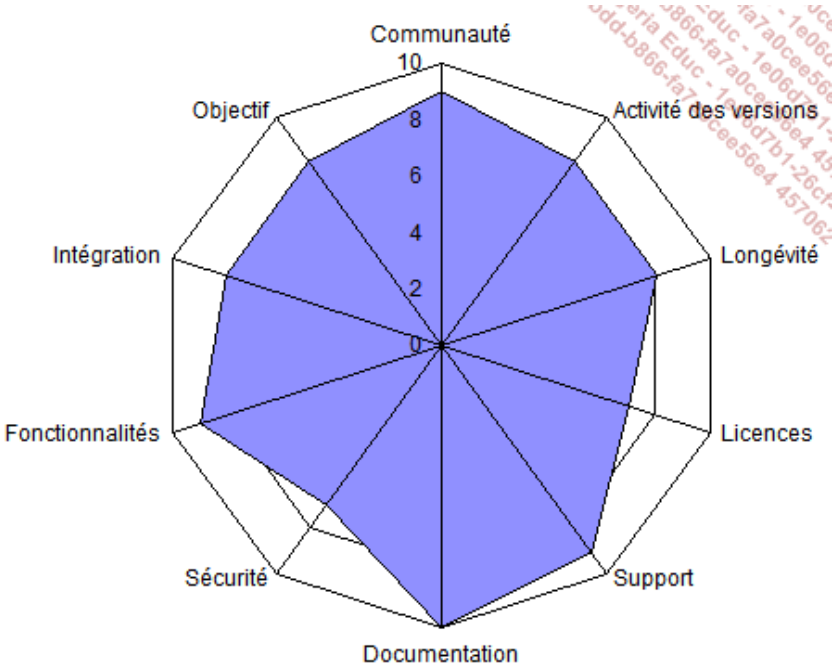
Il ne s'agit pas d'un script en PHP. Il vous suffit d'une feuille Excel classique avec l'ensemble des critères exposés ici. Dès que vous avez noté chaque critère sur 10, il vous suffit de produire un graphique de type Radar.

Parfois la meilleure des solutions ne se trouve pas dans notre camp. Il faut se l'admettre et évoluer. Prendre en compte les avantages d'un autre produit, c'est aussi se permettre de progresser plus rapidement.

d. Exemple avec PHP

PHP	
Communauté	9
Activité des versions	8
Longévité	8
Licences	7
Support	9
Documentation	10
Sécurité	7
Fonctionnalités	9
Intégration	8
Objectif	8

© ENI Editions - All rigths reserved - Algeria Educ



Ici, un simple graphique en radar offre une vue synthétique et globale permettant d'invoquer une juste vérité : le langage PHP est un langage de qualité fédéré autour d'un vrai projet et ayant une vitalité le plaçant parmi les langages de programmation les plus populaires du Web.

© ENI Editions - All rigths reserved - Algeria Educ

Le paradigme du tout spécifique

1. L'illusion de la maîtrise totale

Le mythe de la maîtrise totale est très tenace. En effet, à part les extensions de base, il est très fréquent de trouver des développeurs utilisant leur propre brique pour gérer un système de cache, un moteur de rendu visuel ou des couches d'abstraction d'accès aux bases de données.

Il est illusoire de penser que si l'on refait tout, on aura une meilleure maîtrise de l'ensemble.

2. Problèmes des développements maison

Les développements maison reposent trop souvent sur un petit groupe d'individus concentrant l'ensemble de la connaissance technique des produits.

Il en résulte de plus gros risques sur des briques fondamentales dans les projets :

- Charge de travail supplémentaire.
- Code spécifique.
- Pas d'expertise technique classique.
- Intégration des compétences plus lente.
- Courbe d'apprentissage plus élevée.
- Bugs plus fréquents sur des briques de base.
- Tests unitaires moins fournis.
- Documentation et support limités.

Il en résulte que les compétences concernant les produits deviennent limités, ce qui fait courir un risque important aux entreprises concernant la pérennité des choix technologies pris. Il n'est pas rare que les logiciels issus de développement entièrement interne finissent à la poubelle dans les 3 à 4 ans suivant leur mise en production. En effet, si les résultats initiaux sont toujours encourageants, produire un logiciel de qualité n'est pas le fruit du hasard mais bien d'un processus long s'appuyant sur la maturité de l'ensemble des différentes parties et des composants logiciels.

3. Le pire cauchemar

La pire chose qui puisse arriver dans un projet utilisant des développements trop spécifiques, internes, non documentés et sans le support de personnes qualifiées et expérimentées, est de devoir faire évoluer ce projet sur la base des versions antérieures (bien sûr non maintenues).

Les bugs abondent et personne, finalement, ne pourra vous aider à vous en sortir car la priorité est donnée aux évolutions.

Ce cauchemar, je l'ai vécu et depuis je suis convaincu que d'autres approches sont plus pertinentes. Même si l'entreprise possède des ressources très importantes pour arriver au même résultat, il n'est pas possible aujourd'hui d'obtenir une qualité supérieure en réalisant tous les développements depuis zéro. Ce n'est pas une approche pertinente techniquement, ni économiquement.

Viser la perfection en vous appuyant sur le travail des autres et la maturité de leurs réalisations ouvrira de meilleures perspectives pour vos applications.

S'il est intéressant de comprendre fondamentalement ce qui est réalisé et comment cela fonctionne, il est souvent plus pertinent de faire évoluer un produit existant. La quantité de travail sera moindre que le travail à fournir pour une réécriture ex nihilo.

© ENI Editions - All rigths reserved - Algeria Educ

Il est donc quasiment toujours plus profitable et intelligent de contribuer à un projet pour vos propres besoins plutôt que de refaire le monde dans votre coin.

© ENI Editions - All rigths reserved - Algeria Educ

Le paradigme hybride intégré/spécifique

Beaucoup de développeurs veulent posséder plus de maîtrise sur le code en réécrivant (en beaucoup d'heures de travail) des composants logiciels. Cependant, l'ère d'Internet a ouvert une multitude de possibilités pour diminuer son temps de développement de manière drastique en intégrant des composants réalisés par des tierces personnes, possédant une documentation solide et une plate-forme de qualification, de tests unitaires et de recette souvent effectués de manière communautaire. Les tests et la maintenance des composants communautaires, de par les heures de travail nécessaires, ne peuvent pas être réalisés par une entreprise souhaitant simplement produire un logiciel répondant à un besoin spécifique.

En résumé, l'intégration de composants permet :

- La réduction des temps de développement.
- La qualité des composants logiciels.
- La qualité de l'architecture logicielle en général.
- Le support accru des communautés.
- La maintenabilité générale plus importante.
- Des ressources moins spécifiques.
- Une montée en compétences plus rapide.
- La possibilité d'expertise.

L'approche est très simple : identifiez les briques techniques pour votre logiciel puis faites une analyse OSMM sans conception afin de déterminer les briques techniques les plus pertinentes dans votre contexte.

Si la brique prend autant de temps à être réécrite en interne que le temps nécessaire à son intégration, il est préférable de l'intégrer car, par ce processus, l'intégration des futures versions sera facilitée par l'expérience déjà acquise et ne nécessitera pas autant de ressources que l'évolution du code et l'ensemble des coûts associés en tests, recettes, intégrations supplémentaires.

Intégrer peut être plus coûteux à court terme mais l'expérience acquise, les gains d'évolutivité et la réduction des coûts de maintenance dans le futur valent largement l'effort de cette tâche.

Il est aussi important de valoriser l'expérience acquise. En effet, l'intégration est un processus, une démarche qui permet d'ouvrir des horizons aux développeurs que nous sommes par l'apprentissage d'adaptation, de compréhension, d'assimilation et finalement d'appropriation du travail d'autres équipes de votre entreprise ou d'un projet libre.

Bref, cela élargira vos compétences en vous rendant capable de souplesse vis-à-vis de la nature des composants à assembler.

© ENI Editions - All rigths reserved - Algeria Educ

Choix du modèle de développement

Quand on doit commencer un projet en PHP, on ne sait pas par quoi commencer.

En effet, entre les solutions clés en main qui promettent la lune, les « frameworks » permettant de développer plus rapidement, les composants spécifiques et les classes de composants que l'on retrouve sur plusieurs sites du Web, il y a de quoi se perdre littéralement.

La première chose à faire est de se poser quelques questions clés afin de mieux comprendre votre besoin et définir la meilleure, sinon la plus optimale, des tactiques pour réaliser votre logiciel dans les meilleurs délais et le meilleur coût, selon les compétences que vous avez, vous-même ou votre équipe.

Voici donc les questions à se poser pour commencer tout projet :

- Quel est le pré-requis du logiciel ?
- De quelles ressources je dispose pour réaliser la mise en œuvre ?
- Quels sont les capacités et le temps d'adaptation des équipes ?
- Est-ce que cela va à contre-courant des projets existants ayant réussi ?
- Est-ce un besoin générique d'un point de vue technique ?
- Est-ce un besoin générique d'un point de vue fonctionnel ?
- Quelles sont les parts de spécifique et de générique dans la demande ?

Voici donc une approche permettant la définition d'un modèle de coût réaliste.

Le bon modèle est toujours le moins coûteux, à court terme et à long terme. Reste à définir comment modéliser le coût de la réalisation et coût de la maintenance.

Il est donc important pour cet exercice d'utiliser un certain nombre d'indicateurs et de les mesurer sur une échelle de 0 à 10. Voici une liste d'une vingtaine d'indicateurs permettant de définir assez rapidement si votre projet « colle » bien à la situation.

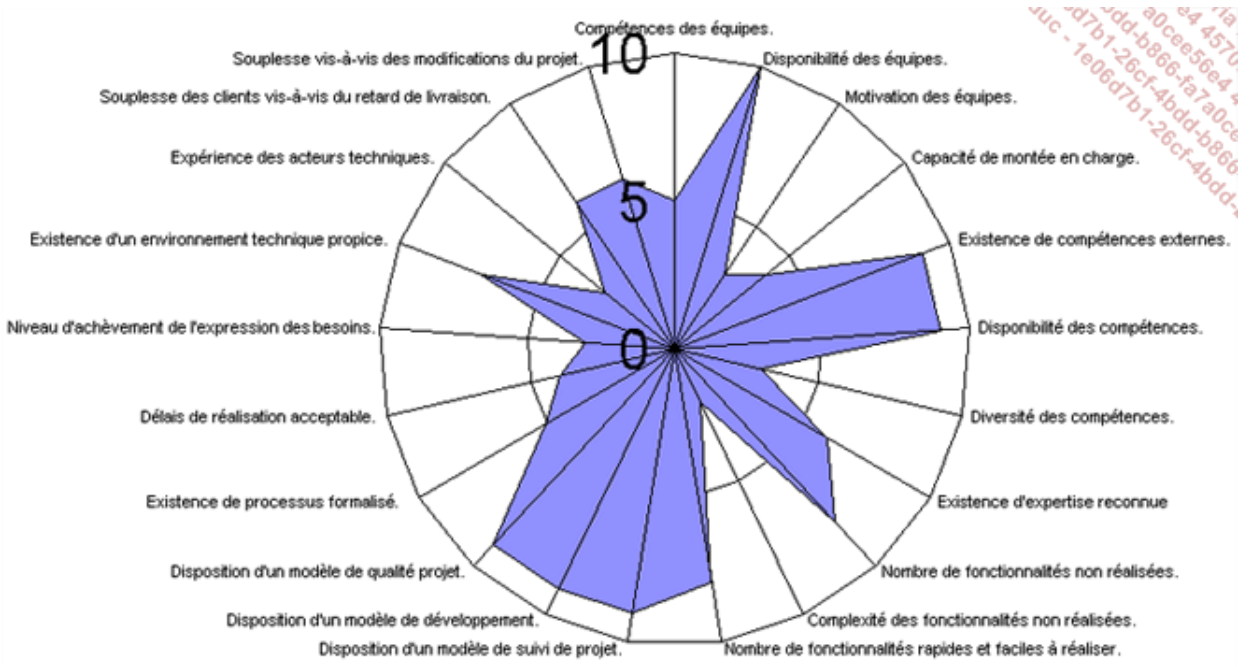
- Compétences des équipes.
- Disponibilité des équipes.
- Motivation des équipes.
- Capacité de montée en charge.
- Existence de compétences externes.
- Disponibilité des compétences.
- Diversité des compétences.
- Existence d'expertise reconnue.
- Nombre de fonctionnalités non réalisées.
- Complexité des fonctionnalités non réalisées.
- Nombre de fonctionnalités rapides et faciles à réaliser.
- Disposition d'un modèle de suivi de projet.

© ENI Editions - All rigths reserved - Algeria Educ

- Disposition d'un modèle de développement.
- Disposition d'un modèle de qualité projet.
- Existence de processus formalisé.
- Délais de réalisation acceptable.
- Niveau d'achèvement de l'expression des besoins.
- Existence d'un environnement technique propice.
- Expérience des acteurs techniques.
- Souplesse des clients vis-à-vis du retard de livraison.
- Souplesse vis-à-vis des modifications du projet.

Sur la base de ces indicateurs, à vous de trouver la meilleure des solutions possibles dans le contexte qui vous concerne.

Le fait de mettre en relief ce type de contenu vous permet de visualiser d'un coup d'œil les forces et les faiblesses d'une solution vis-à-vis d'une autre.



© ENI Editions - All rigths reserved - Algeria Educ

Solutions complètes prêtes à l'emploi

Content Management System Made Simple ou CMSMS est une solution prête à l'emploi idéale pour vos projets Web. CMSMS est une implémentation de CMS. Un CMS permet de gérer la relation entre le contenu et la présentation en faisant pour vous le lien entre contenu et présentation. Pour cela, il offre un certain nombre d'options permettant l'administration de données de présentation et de travail collaboratif.

Dans notre exemple, nous avons choisi CMSMS car la prise en main est rapide et qu'il est extensible à souhait.

1. Principe de CMSMS

Le principe du CMSMS est de rendre les choses simples afin de permettre au plus grand nombre de travailler de manière collaborative le plus rapidement et le plus aisément possible.

Intégrant dès l'installation de nombreux modèles de présentation et de nombreuses données par défaut, il offre la possibilité d'étendre ses fonctionnalités sans passer par des outils d'installation supplémentaires.

2. Fondamentaux de CMSMS

CMSMS se distingue par plusieurs points des autres solutions. Notamment sur trois aspects majeurs :

Le premier d'entre eux est que CMSMS est un assemblage de composants fiables. N'intégrant pas de multiples solutions ni de multiples versions, il offre la possibilité de pleinement exploiter les fonctionnalités de chacune de ses briques de base sans avoir à réduire le potentiel de chacune des briques au regard du potentiel d'une autre brique concurrente.

Le second aspect est que CMSMS possède une architecture modulaire et administrable. En effet, l'ensemble des fonctionnalités s'appuie sur quelques modules fondamentaux. L'ajout d'autres fonctionnalités est réalisé par l'ajout de modules. L'ensemble de ces modules peut être installé par téléchargement depuis l'interface d'administration elle-même.

Le troisième est que CMSMS est extensible et facile à maintenir. Il est possible d'y ajouter ses propres fonctionnalités, soit en réalisant un module, soit en ajoutant des tags utilisateur. Les tags utilisateur étant de petits ensembles de code PHP permettant de réaliser une tâche très simple à l'intérieur d'une page. L'intégration du tag dans une page provoque l'exécution du code PHP associé et l'affichage du résultat.

Ainsi pour ces trois raisons, CMSMS est une bonne base de développement de site Web, pour tout type d'activité, et permet aussi ensuite d'évoluer naturellement vers des applications de plus en plus complexes réalisées sous forme de modules.

3. 2 Briques du CMSMS

CMSMS est divisé en deux parties complètement séparées : le site Web et l'interface d'administration.

a. La partie site Web

Il s'agit de la partie offerte en accès libre sur Internet pour un site Web ou de la partie accessible via Internet pour votre application.

Cette partie est souvent en consultation seule mais peut aussi proposer la gestion des utilisateurs et de groupes d'utilisateurs.

b. La partie administration

Il s'agit de la partie administration des données et de la présentation. Cette interface offre la possibilité de gérer le contenu du site, d'administrer le rendu du site, de gérer les modules et de gérer les accès des administrateurs de tout genre.

© ENI Editions - All rigths reserved - Algeria Educ

Exemples de composants intégrables

Il existe de nombreux domaines où il est impératif de ne pas réécrire des composants existants. Nous pouvons en citer une liste non négligeable :

1. Les clients et les serveurs des protocoles réseau standard

Il n'est pas envisageable de récrire aujourd'hui, ni un client, ni un serveur d'un protocole réseau. En effet, une étude de pertinence permettra rapidement d'identifier des briques réalisant déjà cette fonctionnalité.

a. Client HTTP

Le PHP offre une extension pour curl. Cependant, vous aurez beaucoup de code à écrire avant de pouvoir envoyer vos requêtes et analyser vos réponses, surtout s'il est impératif de réaliser plusieurs échanges tels que s'authentifier puis naviguer sur plusieurs pages, par exemple.

b. Serveur HTTP

Le PHP offre une extension pour gérer des connexions TCP et donc permettre d'écrire son serveur. Cependant, il vous faudra écrire de nombreuses lignes de code avant de pouvoir gérer les appels, l'authentification et surtout les appels simultanés de plusieurs clients HTTP en parallèle.

Les notions de robustesse, de performance, le bon niveau de fonctionnement et les temps de réponse seront des facteurs clés. Le développement spécifique sera donc très coûteux et donc à exclure par défaut.

2. Les composants orientés données

a. Système de cache

Le PHP, par défaut, propose un certain nombre d'extensions pour construire ou utiliser un système de cache.

La gestion du cache est une partie critique dans la maîtrise des performances d'une application. En effet, il est impératif de comprendre le contexte de votre application pour en déterminer le bon système de cache.

Il existe trois types de caches de base :

- Cache en session utilisateur.
- Cache partagé sur un serveur entre différentes sessions.
- Cache distribué entre plusieurs serveurs et utilisateurs.

Parmi toutes les solutions, nous pouvons citer :

- Le serveur Memcached et L'API memcached de PHP.
- Le module PEAR Cache_Lite permettant de mettre en œuvre un cache local.
- Le module Zend ::Cache permettant de réaliser une interface de haut niveau.

b. Gestion des bases de données

La gestion de l'accès aux données est aussi critique pour votre application. En effet, l'utilisation des briques telles que l'extension PDO ou les extensions natives aux bases de données ne va pas vous faciliter la tâche car il sera impératif de coder par vos propres moyens la relation entre votre modèle relationnel et votre modèle objet en PHP. Il s'agit ici de trouver un outil permettant de faciliter les liens entre vos objets en PHP et vos données en base.

© ENI Editions - All rigths reserved - Algeria Educ

Un bon outil de translation (ou mapping) est une des clés du succès pour votre projet et permet l'évolutivité de votre application tout en réduisant le temps d'adaptation des ponts SQL/objet.

Parmi les solutions simples pouvant être mises en place, nous citons les suivantes :

- Le module PEAR DB_DatObject offrant la gestion du schéma automatique.
- Le module Zend ::DB offrant un moyen de récupérer les données sous forme d'objets.

Il existe aussi des outils intégrables n'étant pas des frameworks complets :

- Propel
- Doctrine
- Phpsimbledb
- Pdomap

Il en existe bien d'autres. Apprenez à en maîtriser un correctement et à devenir un utilisateur expert.

c. Gestions des formats de fichiers

L'un des aspects majeurs des applications est leur nécessité à gérer différents formats et à tenter des transformations afin d'obtenir un format unique permettant un traitement sur le même plan de données.

Pour cela, il est possible d'écrire son propre convertisseur de données à chaque fois ou de bénéficier de l'expérience d'autrui en s'appuyant sur des classes préexistantes, maintenues et documentées.

Le fait d'intégrer à ce moment le bon code ne vous fera pas gagner énormément de temps la première fois. L'intégration vous permettra de gagner immédiatement en qualité globale par utilisation de brique stable, fonctionnelle et testée. Puis dans un second temps, vous serez capable de fournir plus rapidement l'intégration de ces composants par le fruit de votre expérience et améliorerez votre efficacité technique.

Il existe de nombreuses classes pour la manipulation des formats de fichiers dans le projet PEAR et PHPClasses.

Il suffit simplement de taper dans les moteurs de recherche le nom du format et vous trouverez des classes pour manipuler des formats tels que :

- Fichier compressé : Zip, Gz, Tar.
- Fichier de données tabulées : CSV, TVS, Excel.
- Fichier de texte : Word, PDF, OpenDocument.
- Fichier de contacts : Vcard, Adress Book.
- Fichier de mot de passe : Apache, Samba.
- Fichier de musique : Mp3, ogg.

d. Gestion des chiffrements de données

Dans le même ordre d'idée, l'utilisation de composants facilitant la gestion de la sécurité des données par chiffrement est préférable et offre un gain de temps immédiat sur l'utilisation, l'évolution et les choix techniques futurs.

La plupart des algorithmes de cryptage et de décryptage se trouvent présents dans l'annuaire des modules PEAR.

e. Utilisation d'un tiers service Web

L'utilisation et l'intégration de tiers service avec des prestataires ne sont pas toujours simples et rapides à mettre

© ENI Editions - All rigths reserved - Algeria Educ

en œuvre. En effet, réussir à faire communiquer correctement et efficacement deux systèmes d'information est un véritable défi. Pour cela, bénéficier de l'expérience d'autrui au travers d'un module PHP bien testé et maintenu est la meilleure des choses à faire.

Ne perdez pas une minute de votre temps à recoder un accès à un service si vous avez découvert une classe faisant ce travail pour vous. De plus, les classes PHP offrent des services supplémentaires comme la reconnexion transparente ou l'enchaînement de plusieurs opérations techniques permettant d'obtenir des fonctionnalités avancées à moindre effort.

Il existe 46 modules PEAR de gestion de services Web.

Le projet PHPClasses offre lui aussi un nombre important de solutions de connexion et d'utilisation des services proposés sur le Web.

Il existe de nombreux services accessibles simplement tels que :

- Service de détection de spam : Askimet.
- Service de stockage : Amazon.
- Service de gestion de blog : WordPress, Blogger, etc.
- Service de bookmark en ligne : Digg, Delicious, Technorati, etc.
- Service de réseaux sociaux : Facebook, Twitter, Google, etc.
- Service d'URL courtes : TinyURL.
- Service générique : REST, XML, SOAP, UDDI.

3. Composants support des fonctionnalités

a. Gestion des paiements

Il existe de nombreux composants gérant facilement le paiement en ligne, et ceci en installant un minimum de fichiers. Les composants de paiement en ligne sont très sensibles car ils mêlent échanges sécurisés, transactions financières et données personnelles. Réaliser de tels modules est très complexe surtout en terme de validation technique.

Ces classes offrent, de fait, une alternative à du code fait main très volumineux.

En effet, chaque module est souvent bien maintenu et documenté. Le fait d'utiliser un module ayant un mainteneur permet de gérer l'évolution et surtout d'avoir un point unique pour offrir si nécessaire les évolutions techniques qui seront réintégrées dans le code.

Voici quelques-unes des solutions prêtes à l'emploi utiles pour le paiement en ligne :

- Les modules PEAR de la section « Payment ».
- La classe PhpPaypal de PhpWeby.
- La classe Paypal Payment Data Transferts.

Il existe ensuite, sur le paiement en ligne, un nombre important de solutions clé en main. Ce sont des produits plus complets qui offrent des fonctionnalités plus larges.

Nous pouvons citer quelques-unes de ces solutions :

- OsCommerce.
- ZenCart.
- Opencart.

© ENI Editions - All rigths reserved - Algeria Educ

- L'ensemble des modules pour CMS ou blog de Ecommerce.

b. Gestion des traces

La gestion des traces est une des clés de la qualité dans les applications PHP. En effet, une application utilisant un module de gestion des traces appropriées pourra offrir la possibilité de changer toute la gestion des traces sans avoir à retoucher une seule ligne de code de votre application. La seule chose à faire sera de paramétrer votre composant de gestion des traces.

Parmi l'ensemble des composants PHP de gestion des traces, LOG4PHP est un bon candidat. Il s'agit d'un portage en PHP d'une brique logicielle de gestion des traces du monde Java.

Cette brique logicielle est intéressante, puissante et maintenue. Elle nécessite un temps d'apprentissage afin de bien maîtriser les fondamentaux et de l'utiliser au mieux pour vos besoins.

Log4PHP possède les fonctionnalités suivantes :

- Configuration au travers d'un fichier XML ou d'un fichier de propriétés.
- Multiples supports de log.
- Multiples gestionnaires de formats de traces.
- Gestionnaire de contexte pour le diagnostic.

Log4PHP permet de supporter les traces dans diverses sources de stockage :

- Fichier de trace.
- Fichier de trace archivé par taille.
- Fichier de trace archivé par jour.
- Ecriture dans le résultat (page HTML par exemple).
- Affichage dans le fichier d'erreurs.
- Envoi de traces par Email.
- Base de données via l'extension PDO.
- Serveur de log Syslog.
- Serveur d'événement NT.
- Serveur TCP générique.

c. Gestion du rendu visuel

L'un des points clés permettant d'architecturer correctement votre application PHP est de séparer l'application en trois couches.

Nous avons vu qu'il existe plusieurs solutions pour effectuer la traduction Objet - relationnel.

Pour le rendu visuel, il existe des nombreux frameworks intégrant des moteurs de rendu visuel. Cependant, ces frameworks ont le problème des applications « trop » structurées, ils sont sous-performantes.

Pour vos besoins, il est impératif d'utiliser un moteur de rendu visuel afin de permettre une évolution de votre code applicatif séparée de celle de ses fonctionnalités et de sa qualité, de la manière dont l'information est échangée avec vos utilisateurs.

© ENI Editions - All rigths reserved - Algeria Educ

Il existe plusieurs moteurs de rendu ou moteurs de modèle (template engine) écrits en PHP, en voici quelques-uns :

- TinyButStrong
    - http://www.tinybutstrong.com/
- Smarty
    - http://www.smarty.net
- Dwoo
    - http://www.dwoo.org/
- HTML_Template_Flexy
    - http://pear.php.net/package/HTML_Template_Flexy
- Sugar
    - http://php-sugar.net/

d. Gestion de tout ce qui prend du temps

En règle générale, il est important de ne jamais investir un iota de votre temps pour réinventer la roue. La plupart des composants sont, si vous les analysez correctement, suffisamment aboutis pour rendre le service de base qu'ils vous proposent avec une qualité de code correct.

La délicate question qui vient ensuite est : que dois-je faire si cela me prend autant de temps d'intégrer une des briques logicielles que de réécrire son code ?

Cette question a une réponse simple et un nombre important de justificatifs.

La réponse consiste à intégrer malgré tout toujours plus de composants. Les raisons sont simples et peuvent être listées rapidement :

- Apprentissage d'un nouveau composant.
- Acquisition du savoir-faire d'intégrateur.
- Possibilité de contribuer à la qualité.
- Evolutivité naturelle du composant indépendant de votre code.
- Gain d'intégration sur les futurs projets.
- Acquisition de l'esprit de travail collaboratif.
- Amélioration du composant par retour d'expérience de votre part.

Bref, le seul avantage de refaire le travail est de comprendre « à quel point » il est complexe de réaliser un bon composant, sans bug fonctionnel, évolutif et performant.

Une seconde question importante à se poser est : Que dois-je vraiment faire si l'intégration du composant prend plus de temps ?

La réponse est moins simple qu'il n'y paraît et doit être mise en relief par rapport aux gains à moyen et long termes.

Quels gains ce type de composant peut-il nous offrir en temps et en qualité sur l'ensemble du projet sur lequel je travaille ?

Il est intéressant de constater que ce type de question offre une autre perspective et relativise fortement la notion

© ENI Editions - All rigths reserved - Algeria Educ

de perte de temps initiale par la mise en évidence de l'intérêt pour l'application dans son ensemble au moment de la remise finale.

Ce type d'approche offre la capacité à long terme de produire des applications PHP de qualité et permet de développer une équipe performante capable de produire plus rapidement des solutions en PHP de bonne qualité.

Il n'y a pas de bonne réponse. Cependant, investir dans l'intégration de composants externes revient à investir dans l'avenir des applications, des équipes, des bonnes pratiques, du niveau de qualité et, finalement, dans la performance globale.

4. Quelques exemples

Pour finir ce chapitre, voici quelques extraits de code de composants intégrés dans divers projets et vous pourrez ainsi juger de la complexité de l'intégration et des services que ceux-ci rendent à leur projet.

a. Composants de rendu visuel

Voici un exemple d'utilisation de Smarty. Smarty est un moteur de rendu visuel ayant deux systèmes de cache intégré. Le premier permet de stocker en cache la compilation du modèle de rendu en script PHP. Le second cache, plus agressif, stocke le résultat de sortie.

Installation de Smarty

```
$ mkdir templates
$ mkdir templates_c
$ mkdir cache
$ mkdir configs
$ tar xzf ~/Smarty-2.6.26.tar.gz
$ ln -s Smarty-2.6.26/libs/ Smarty
$ chmod 777 templates cache/  templates_c/
$ $ php index.php
Bonjour Jean-Marie Voici le résultat Smarty!
```

Exemple de modèle Smarty

```
{config_load file="global.ini" section='Ami'}



Bonjour, {$nom}, Bienvenue sur mon premier rendu visuel Smarty !



<table border='1' width="{#largeur#}">
    <tr bgcolor="{#couleurEntete#}"><th>Nom</th><th>Age</th></tr>

    {foreach from=$ages key=ami item=age}
           <tr><td>{$ami}</td><td>{$age}</td></tr>
    {/foreach}
</table>
```

Exemple de code d'utilisation de Smarty

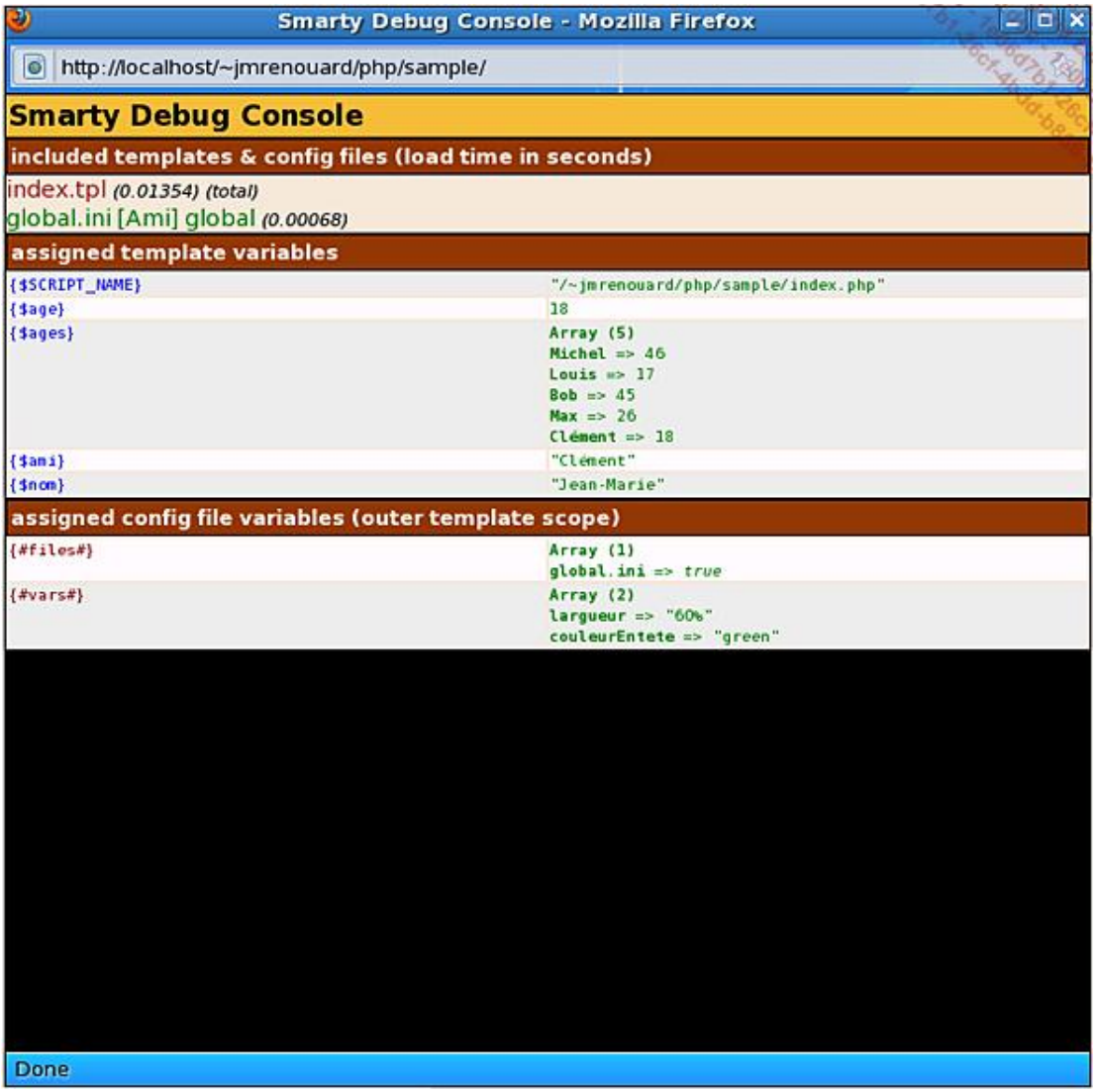
```
<?php

require_once('Smarty/Smarty.class.php');

# création de l'objet Smarty
$smarty = new Smarty();

# positionnement du répertoire des templates
$smarty->template_dir = 'templates/';
```


© ENI Editions - All rigths reserved - Algeria Educ

```
# positionnement du répertoire des templates compilés
$smarty->compile_dir  = 'templates_c/';

# positionnement du répertoire des configurations
$smarty->config_dir   = 'configs/';

# positionnement du répertoire des caches de résultats
$smarty->cache_dir    = 'cache/';

# Ajout d'une variable dans le template
$smarty->assign('nom','Jean-Marie');

$tableAge = array(
    "Michel "=> 46,
    "Louis" =>17,
    "Bob" => 45,
    "Max" => 26,
    "Clément" => 18,
    );


# ajout d'un tableau PHP dans le template
$smarty->assign('ages', $tableAge);


# Activation du cache de résultat de template
# Le résultat final est stocké et sera redistribué sans
recompilation du template ni execution de celui-ci.
# Cache à durée d'expiration individuelle
$smarty->caching = 2;

# durée d'expiration positionnée à 3 minutes (180 sec.)
$smarty->cache_lifetime = 180;

# Activation de la fenêtre pop-up de debuggage
$smarty->debugging = true;

# Demande de restitution du template index.tpl
$smarty->display('index.tpl');
?>
```

Résultat de l'exécution du code partie debug

© ENI Editions - All rigths reserved - Algeria Educ



Résultat de l'exécution du code

Bonjour, Jean-Marie, Bienvenue sur mon premier rendu visuel Smarty !

Nom	Age
Michel	46
Louis	17
Bob	45
Max	26
Clément	18

b. Bibliothèque PEAR

La bibliothèque PEAR est un répertoire de composants en PHP, installable rapidement. Chaque composant est packagé de manière normalisée facilitant le déploiement, le test et l'intégration.

Le code respecte des normes de codage strictes permettant d'homogénéiser le code de l'ensemble des modules.

Voici un exemple d'utilisation de PEAR dans le cadre d'une implémentation d'un gestionnaire d'événements.

L'idée est qu'il existe une usine fabriquant des voitures. À chaque création d'un véhicule, l'automobile doit être vérifiée par une équipe de qualité et éventuellement refusée et détruite.

Installation du module PEAR Event_Dispatcher

```
#pear install Event_Dispatcher
downloading Event_Dispatcher-1.1.0.tgz ...
```


© ENI Editions - All rigths reserved - Algeria Educ

```
Starting to download Event_Dispatcher-1.1.0.tgz (8,500 bytes)
.....done: 8,500 bytes
install ok: channel://pear.php.net/Event_Dispatcher-1.1.0
```

```
<?php

class Automobile {
    private $couleur='noir';
    private $estConforme=true;
    private $raison="OK";

    public function __construct($c='noir') {
        $this->couleur=$c;
    }

    public function getCouleur() {
        return $this->couleur;
    }
    public function demarrer() {
        echo "\n\t* Je démarre...";
    }

    public function rouler() {
        echo "\n\t* Je roule...";
    }

    public function arreter() {
        echo "\n\t* Je m'arrête...";
    }
    public function getEstConforme() {
        return $this->estConforme;
    }
    public function setEstConforme($e, $raison='Fonctionne
superbe') {
        $this->estConforme=$e;
        $this->raison=$raison;
    }
    public function getMessageControle() {
        return $this->raison;
    }
}
?>
```

Classe AutomobileEventDispatcher

```
<?php

class AutomobileEventDispatcher extends Automobile{
    private $dispatcher;

    public function __construct($d, $c='noir') {
        $this->dispatcher=$d;
        parent::__contruct($c);

    }

    public function getDispatcher() {
        return $this->dispatcher;
     }

     public function demarrer() {
          parent::demarrer();
          $n=$this->dispatcher->post($this, 'demarrage');
          if ($n->isNotificationCancelled()) {
            $this->setEstConforme(FALSE, "Auto ne démarre pas
correctement.");
        }
```

```
     }

     public function rouler() {
          parent::rouler();
          $n=$this->dispatcher->post($this, 'deplacement');
          if ($n->isNotificationCancelled()) {
            $this->setEstConforme(FALSE, "Auto ne se deplace pas
bien.");
        }
     }

     public function arreter() {
          parent::arreter();
          $n=$this->dispatcher->post($this, 'arret');
          if ($n->isNotificationCancelled()) {
            $this->setEstConforme(FALSE, "Auto ne s'arrête pas
bien.");
        }
     }
}
?>
```

Classe ControleurQualité

```
<?php

class ControleurQualite {
     private $nom;
     private $proba=10;
     public function __construct($n) {
          $this->nom=$n;
     }
     public function getNom() {
          return $this->nom;
     }

     public function controleDemarrage(&$notification) {
          $auto = &$notification->getNotificationObject();
          echo "\n\t\t" . $this->getNom() . " controle le
demarrage.";
          if ( rand(1, $this->proba) == $this->proba) {
               echo "\n\t\t" . $this->getNom() . " ça démarre
pas bien.";
               $notification->cancelNotification();
          } else {
               echo "\n\t\t" . $this->getNom() . " ça démarre
super.";
          }
     }
    public function controleRouler(&$notification) {
          $auto = &$notification->getNotificationObject();
          echo "\n\t\t" . $this->getNom() . " contrôle le
déplacement.";
          if ( rand(1, $this->proba) == $this->proba) {
               echo "\n\t\t" . $this->getNom() . " ça roule
mal.";
               $notification->cancelNotification();
          } else {
               echo "\n\t\t" . $this->getNom() . " ça roule
bien.";
          }
     }

     public function controleArret(&$notification) {
          $auto = &$notification->getNotificationObject();
          echo "\n\t\t" . $this->getNom() . " controle l'arrêt.";
          if ( rand(1, $this->proba) == $this->proba) {
               echo "\n\t\t" . $this->getNom() . " ça ne
```


© ENI Editions - All rigths reserved - Algeria Educ

```
s’arrête pas du tout.";
                $notification->cancelNotification();
          } else {
                echo "\n\t\t" . $this->getNom() . " ça s’arrête
correctement.";
          }
     }
}
?>
```

Code de gestion de l’usine

```
<?php

// Rapporte les erreurs d’exécution de script
error_reporting(E_ERROR | E_WARNING | E_PARSE);

require_once ’Event/Dispatcher.php’;

function __autoload($class_name) {
    require_once $class_name . ’.class.php’;
}

$dispatcher = &Event_Dispatcher::getInstance();

// Georges se charge de démarrer les voitures et contrôle le
démarrage
$controleur1=new ControleurQualite("Georges");

// Gaston se charge de rouler avec les voitures et contrôle le
déplacement
$controleur2=new ControleurQualite("Gaston");

// Grégoire arrête les voitures et contrôle l’arrêt effectif
$controleur3=new ControleurQualite("Grégoire");


$dispatcher->addObserver(array($controleur1, ’controleDemarrage’),
’demarrage’);
$dispatcher->addObserver(array($controleur2, ’controleRouler’),
’deplacement’);
$dispatcher->addObserver(array($controleur3, ’controleArret’),
’arret’);

$a=new AutomobileEventDispatcher($dispatcher, "Jaunes");

$a->demarrer();
$a->rouler();
$a->arreter();

echo "\n* la voiture Jaune est ".
($a->getEstConforme()?"CONFORME":"NON CONFORME").
" - ".
$a->getMessageControle();
?>
```

Résultat de quelques exécutions

```
$php UsineAutomobile.php
        * Je démarre...
                Georges contrôle le démarrage.
                Georges ça démarre super.
        * Je roule...
                Gaston contrôle le déplacement.
                Gaston ça roule bien.
        * Je m’arrête...
                Grégoire contrôle l’arrêt.
```

© ENI Editions - All rigths reserved - Algeria Educ

```
                Grégoire ça s'arrête correctement.
* la voiture 1 est CONFORME - OK
```

```
$php UsineAutomobile.php
        * Je démarre...
                Georges contrôle le démarrage.
                Georges ça démarre pas bien.
        * Je roule...
                Gaston contrôle le déplacement.
                Gaston ça roule bien.
        * Je m'arrête...
                Grégoire contrôle l'arrêt.
                Grégoire ça s'arrête correctement.
* la voiture 1 est NON CONFORME - Auto ne démarre pas
correctement.
```

```
$php UsineAutomobile.php

        * Je démarre...
                Georges contrôle le démarrage.
                Georges ça démarre super.
        * Je roule...
                Gaston contrôle le déplacement.
                Gaston ça roule bien.
        * Je m'arrête...
                Grégoire contrôle l'arrêt.
                Grégoire ça ne s'arrête pas du tout.
* la voiture 1 est NON CONFORME - Auto ne s'arrête pas bien.
```

c. Cas de client HTTP

Le cas du client HTTP est le plus courant de tous les composants intégrables par défaut. En effet, s'il n'est pas difficile de créer un code fonctionnant rapidement avec le protocole HTTP, cela se gâte quand vous devez utiliser un relais HTTP dans votre entreprise, quand vous devez sécuriser via des certificats SSL les connexions, quand vous devez assurer la continuité des accès et la reprise sur erreur ou quand vous devez offrir un ensemble de connexions permanentes pour plusieurs clients.

Si la gestion d'une connexion HTTP est très simple, gérer un nombre important de connexions HTTP dans un environnement sollicité demande la réalisation de code de qualité.

Pour simplifier votre démarche de connexion, vous pouvez utiliser le composant http_Client du projet PEAR.

Ici, le code suivant permet de stocker dans un fichier le contenu d'une URL quelconque présente sur le Web en passant par un proxy Web et en suivant les redirections automatiquement jusqu'à 5 fois.

Installation de HTTP_Client

```
# pear install Http_Client
downloading HTTP_Client-1.2.1.tgz ...
Starting to download HTTP_Client-1.2.1.tgz (10,202 bytes)
.....done: 10,202 bytes
downloading HTTP_Request-1.4.4.tgz ...
Starting to download HTTP_Request-1.4.4.tgz (17,109 bytes)
...done: 17,109 bytes
downloading Net_URL-1.0.15.tgz ...
Starting to download Net_URL-1.0.15.tgz (6,303 bytes)
...done: 6,303 bytes
downloading Net_Socket-1.0.9.tgz ...
Starting to download Net_Socket-1.0.9.tgz (5,173 bytes)
...done: 5,173 bytes
install ok: channel://pear.php.net/Net_Socket-1.0.9
install ok: channel://pear.php.net/Net_URL-1.0.15
install ok: channel://pear.php.net/HTTP_Request-1.4.4
install ok: channel://pear.php.net/HTTP_Client-1.2.1
```

Exemple d'intégration de code HTML distant

© ENI Editions - All rigths reserved - Algeria Educ

```
<?php

require_once('HTTP/Client.php');

$params=array();
$params['proxy_host'] = 'xxx.xxx.xxx.xxx';
$params['proxy_port'] = 3128;

$headers=array( "User-Agent" => "Mozilla Firefox/3.6.6");
$client= new HTTP_Client( $params, $headers );
$client->setMaxRedirects(5);


$client->enableHistory(false);

$reponse='';
$ret= $client->get($argv[1], $reponse);

echo "Code HTTP: $ret\n";

if ($ret == 200) {
     $rep=$client->currentResponse();

     if ( isset($argv[2]) ) {
          file_put_contents($argv[2], $rep['body']);
     } else {
          print_r($rep);
     }
} else {
      echo "Erreur requete HTTP\n";
}
?>
```

d. Cas de l'accès aux données en base de données

Pour notre exemple, nous présenterons l'utilisation de PEAR DB_DatObject dans une relation entre un livre et son auteur.

Le but est de pouvoir lire et écrire des données directement depuis le code PHP sans gérer explicitement le code SQL.

Installation composant PEAR

```
# pear install DB_DataObject
Did not download optional dependencies: pear/MDB2, pear/Validate,
use --alldeps to download automatically
pear/DB_DataObject can optionally use package "pear/MDB2" (version
>= 2.0.0RC1)
pear/DB_DataObject can optionally use package "pear/Validate"
(version >= 0.1.1)
downloading DB_DataObject-1.9.5.tgz ...
Starting to download DB_DataObject-1.9.5.tgz (70,745 bytes)
................done: 70,745 bytes
downloading DB-1.7.13.tgz ...
Starting to download DB-1.7.13.tgz (132,246 bytes)
...done: 132,246 bytes
downloading Date-1.4.7.tgz ...
Starting to download Date-1.4.7.tgz (55,754 bytes)
...done: 55,754 bytes
install ok: channel://pear.php.net/Date-1.4.7
install ok: channel://pear.php.net/DB-1.7.13
install ok: channel://pear.php.net/DB_DataObject-1.9.5

# pear install MDB2
downloading MDB2-2.4.1.tgz ...
Starting to download MDB2-2.4.1.tgz (119,790 bytes)
.....done: 119,790 bytes
```

© ENI Editions - All rigths reserved - Algeria Educ

```
install ok: channel://pear.php.net/MDB2-2.4.1
MDB2: Optional feature fbsql available (Frontbase SQL driver for
MDB2)
MDB2: Optional feature ibase available (Interbase/Firebird driver
for MDB2)
MDB2: Optional feature mysql available (MySQL driver for MDB2)
MDB2: Optional feature mysqli available (MySQLi driver for MDB2)
MDB2: Optional feature mssql available (MS SQL Server driver for
MDB2)
MDB2: Optional feature oci8 available (Oracle driver for MDB2)
MDB2: Optional feature pgsql available (PostgreSQL driver for
MDB2)
MDB2: Optional feature querysim available (Querysim driver for
MDB2)
MDB2: Optional feature sqlite available (SQLite2 driver for MDB2)
To install use "pear install pear/MDB2#featurename"

# pear install MDB2#mysql
Skipping package "pear/MDB2", already installed as version 2.4.1
downloading MDB2_Driver_mysql-1.4.1.tgz ...
Starting to download MDB2_Driver_mysql-1.4.1.tgz (36,481 bytes)
..........done: 36,481 bytes
install ok: channel://pear.php.net/MDB2_Driver_mysql-1.4.1
```

Code SQL de création des tables

```
DROP TABLE IF EXISTS AUTEUR;
CREATE TABLE AUTEUR (
     id INTEGER PRIMARY KEY,
     nom VARCHAR(255)
);

INSERT INTO AUTEUR  VALUES (1, ’Georges’);
INSERT INTO AUTEUR  VALUES (2, ’Gregoire’);
INSERT INTO AUTEUR  VALUES (3, ’Gaston’);
DROP TABLE IF EXISTS LIVRE;
CREATE TABLE LIVRE (
     id int primary key,
     titre VARCHAR(255),
     auteurid INTEGER
);

INSERT INTO LIVRE VALUES (1, "Georges autobiographie", 1);
INSERT INTO LIVRE VALUES (2, "Ma femme", 1);
INSERT INTO LIVRE VALUES (3, "Mes enfants", 1);
INSERT INTO LIVRE VALUES (4, "Grégoire autobiographie", 2);
INSERT INTO LIVRE VALUES (5, "Mon oeuvre", 3);
INSERT INTO LIVRE VALUES (6, "Ma vie", 3);
```

Code de transformation SQL/PHP via DB_DataObject

```
<?php

require_once(’PEAR.php’);
require_once(’DB/DataObject.php’);

$options = &PEAR::getStaticProperty(’DB_DataObject’,’options’);
$options["database"] =  "mysql://auteur:auteur@localhost/auteur";
$options["proxy"] = "full";

DB_DataObject::debugLevel(0);
class DataObject_Livre extends DB_DataObject {
      public $__table="LIVRE";
      public $id;
      public $titre;
      public $auteurid;
}
```


© ENI Editions - All rigths reserved - Algeria Educ

```
class DataObject_Auteur extends DB_DataObject {
      public $__table="AUTEUR";
      public $id;
      public $nom;

      public function getLivres() {
     $lvr = new DataObject_Livre();

     $lvr->setauteurid($this->id);
     $lvr->find();

     $livres = array();
     while ($lvr->fetch()) {
              $livres[] = clone($lvr);
     }
     return $livres;
      }
}

?>
```

Code d'utilisation du transformateur

```
<?php

require_once('Modele.php');

$auteur = new DataObject_Auteur();
echo "Nb Auteur: ". $auteur->count();

echo "\n";
$auteur->get('id', 1);
print_r($auteur);

$auteur2 = new DataObject_Auteur();
$auteur2->find();

$auteurs = array();
while ($auteur2->fetch()) {
    $auteurs[] = clone($auteur2);
    echo " \n* $auteur2->id )  $auteur2->nom";
    foreach($auteur2->getLivres() as $livre) {
    echo "\n\t* $livre->titre";
    }
}
?>
```

Résultat de l'exécution

```
Nb Auteur: 3
DataObject_Auteur Object
(
    [__table] => AUTEUR
    [id] => 1
    [nom] => Georges
    [_DB_DataObject_version] => 1.9.5
    [N] => 1
    [_database_dsn] =>
    [_database_dsn_md5] => 6655ead300bdccab56852ee935d1af21
    [_database] => auteur
    [_query] => Array
         (
             [condition] =>
             [group_by] =>
             [order_by] =>
             [having] =>
```

© ENI Editions - All rigths reserved - Algeria Educ

```
            [limit_start] =>
            [limit_count] =>
            [data_select] => *
            [unions] => Array
                    (
                    )

        )

     [_DB_resultid] => 2
     [_resultFields] =>
     [_link_loaded] =>
     [_join] =>
     [_lastError] =>
)

* 1 ) Georges
         * Georges autobiographie
         * Ma femme
         * Mes enfants
* 2 ) Gregoire
         * Grégoire autobiographie
* 3 ) Gaston
         * Mon oeuvre
         * Ma vie
```

e. Cas d'utilisations conjointes de DB_DataObject et Smarty

Modèle Smarty

```
{config_load file="global.ini" section='Ami'}


<h1>{$count} Auteurs en base.</h1>


<table border='1' width="{#largeur#}">
    <tr
bgcolor="{#couleurEntete#}"><th>Auteur</th><th>Livres</th></tr>

    {foreach from=$auteurs key=id item=auteur}
           <tr>
                <td>{$auteur}</td>
                <td>
                    <ul>
                    {foreach from=$livres[$id] item=titre}
                        <li>{$titre}</li>
                    {/foreach}
                    </ul>
                </td>
           </tr>
    {/foreach}
</table>
```

Code d'utilisation du transformateur et de Smarty

```
<?php

require_once('Smarty/Smarty.class.php');
require_once('Modele.php');

# création de l'objet Smarty
$smarty = new Smarty();

# positionnement du répertoire des templates
$smarty->template_dir = 'templates/';
```


© ENI Editions - All rigths reserved - Algeria Educ

```
# positionnement du répertoire des templates compilés
$smarty->compile_dir  = 'templates_c/';

# positionnement du répertoire des configurations
$smarty->config_dir   = 'configs/';

# positionnement du répertoire des caches de résultats
$smarty->cache_dir    = 'cache/';

# Ajout d'une variable dans le template

$smarty_auteur=array();

$auteur = new DataObject_Auteur();

$smarty->assign('count', $auteur->count());
$auteur = new DataObject_Auteur();
$auteur->find();

$smarty_auteur['auteurs']=array();
while ($auteur->fetch()) {
    $smarty_auteur['auteurs'][$auteur->id]=$auteur->nom;
           $smarty_auteur['auteur_livre'][$auteur->id]=array();
    foreach($auteur->getLivres() as $livre) {
    array_push($smarty_auteur['auteur_livre'][$auteur->id],
$livre->titre);
    }
}

$smarty->assign('auteurs', $smarty_auteur['auteurs']);
$smarty->assign('livres', $smarty_auteur['auteur_livre']);
# Cache à durée d'expiration individuelle
$smarty->caching = 2;

# durée d'expiration positionnée à 3 minutes (180 sec.)
$smarty->cache_lifetime = 180;

# Activation de la fenêtre pop-up de debuggage
$smarty->debugging = true;

# Demande de restitution du template index.tpl
$smarty->display('auteur.tpl');

?>
```

Résultat de la console de debug

© ENI Editions - All rigths reserved - Algeria Educ

Smarty Debug Console - Mozilla Firefox

http://localhost/~jmrenouard/php/sample/exemple.php

3 Auteurs en base.

Auteur	Livres
Georges	• Georges autobiographie • Ma femme • Mes enfants
Grégoire	• Grégoire autobiographie
Gaston	• Mon oeuvre • Ma vie

Smarty Debug Console

included templates & config files (load time in seconds)

auteur.tpl *(0.00271) (total)*

assigned template variables

{$SCRIPT_NAME}	"/~jmrenouard/php/sample/exemple.php"
{$auteurs}	Array (3) 1 => "Georges" 2 => "Gregoire" 3 => "Gaston"
{$count}	3
{$livres}	Array (3) 1 => Array (3) 0 => "Georges autobiographie" 1 => "Ma femme" 2 => "Mes enfants" 2 => Array (1) 0 => "Gregoire autobiographie" 3 => Array (2) 0 => "Mon oeuvre" 1 => "Ma vie"

assigned config file variables (outer template scope)

{#files#}	Array (0)
{#vars#}	Array (0)

Done private FoxyProxy: Disabled

Résultat d'exécution

3 Auteurs en base.

Auteur	Livres
Georges	• Georges autobiographie • Ma femme • Mes enfants
Grégoire	• Grégoire autobiographie
Gaston	• Mon oeuvre • Ma vie

© ENI Editions - All rigths reserved - Algeria Educ

Qu’est-ce que la haute disponibilité ?

Le concept haute disponibilité est la capacité d’un système d’information de garantir un service ou un traitement d’information et ceci, quelles que soient les pannes ou l’indisponibilité des services tiers. Le concept de haute disponibilité est souvent associé à celui de montée en charge ou « scalability ». Ce dernier a pour objectif de permettre de garantir l’absorption d’une plus grande quantité de trafic par l’ajout de ressources matérielles supplémentaires.

En effet, il est souvent avéré qu’un logiciel fonctionne très bien jusqu’à un certain niveau de sollicitation et qu’une fois ce point (nommé point de rupture) atteint, le service se dégrade fortement et consomme rapidement une grande partie des ressources du système.

À ce stade, seuls un audit approfondi du code ainsi que des tests de tenue en charge permettront de valider ou non la capacité d’une solution logicielle de tenir une montée en charge par ajout de ressources physiques (serveur, réseau, espace de stockage, etc.).

Cela signifie en clair que les équipements physiques doivent être au minimum doublés.

Voici une liste exhaustive des équipements nécessaires à la haute disponibilité :

- Alimentation électrique du site.
- Climatisation.
- Pare-feu.
- Routeur réseau.
- Équipement de réparation de trafic.
- Serveur physique.

La mise en place de la haute disponibilité, de bout en bout, est donc une démarche qui nécessite des moyens financiers et humains importants. Cependant, l’apparition de centres d’hébergement a permis de réduire les investissements en équipements électriques et réseau tout en conservant la possibilité de gérer une architecture système et réseau autonome.

Ce chapitre propose un ensemble de moyens de mettre en place des systèmes de haute disponibilité pour vos applications en s’appuyant sur des solutions intégrées et simples à mettre en œuvre.

Le concept de haute disponibilité n’est pas un concept de sécurité, quoique très proche. Son objectif est de prévenir et de minimiser les impacts d’une défaillance interne d’équipements physiques ou de l’utilisation intensive du service.

Du point de vue de la sécurité, le but est d’empêcher, par tous les moyens, l’utilisation inappropriée d’un service. Du point de vue de la haute disponibilité, le but est de garantir une réponse de qualité et appropriée pour tous les utilisateurs du service, et ceci, quel que soit le type de sollicitation du service aussi bien en matière de volume d’utilisation qu’en matière de nature de l’utilisation des services.

1. Type de répartition de trafic

Il existe deux types de haute disponibilité :

- Les architectures montées en mode normal - secours.
- Les architectures montées en mode nominal - nominal.

Les premières sont simples à mettre en œuvre et nécessitent au minimum la mise en place de mécanismes de redondance d’information souvent limités au niveau des bases de données.

Les secondes nécessitent une redondance à plusieurs niveaux et, souvent même, une adaptation des logiciels afin de permettre le fonctionnement en haute disponibilité totale.

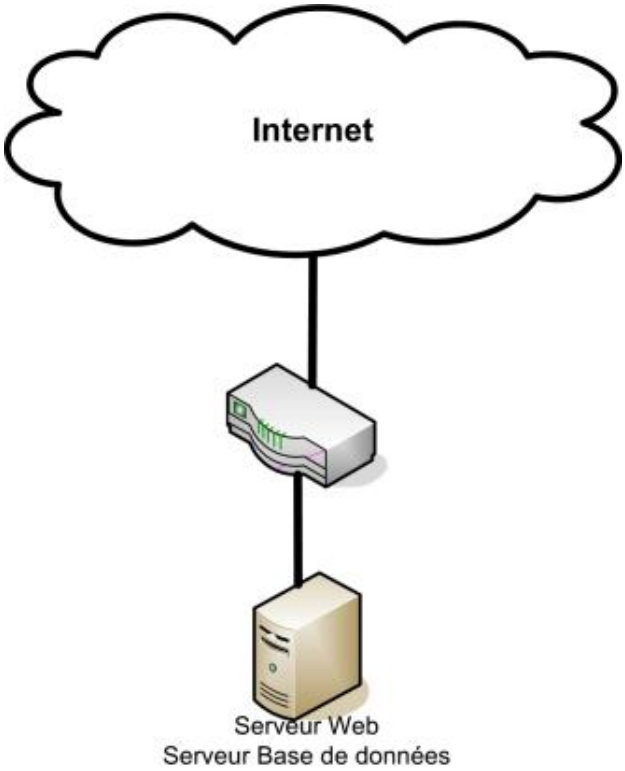
2. Présentation d’un exemple classique

© ENI Editions - All rigths reserved - Algeria Educ

Pour commencer, voici un exemple de redondance de bout en bout : l'étude d'un site Web et d'une base de données que nous souhaitons mettre en place afin de garantir une haute disponibilité du service.

Imaginons simplement un serveur unique avec sa base de données en local.

Du point de vue de la performance d'accès à la base de données, cette solution est idéale. En effet, pas de problématique de type coupure réseau ni de type latence réseau rendant l'échange entre deux serveurs parfois long. Cette temporisation de réponse d'un serveur entraîne automatiquement une dégradation de la qualité du service par l'augmentation de la durée de réponse du service tiers.



D'un autre point de vue, cette solution possède de nombreux inconvénients.

Le premier est que tout repose sur une seule machine. Si la machine est éteinte, il n'y a plus de service du tout (réalité « physique »). Si le serveur tombe en panne, il doit être changé ou réparé et, pendant le temps de maintenance, il est donc impossible de rendre un quelconque service.

Deuxièmement, il est difficile de cibler l'origine des problèmes de performance d'un serveur en fonction des services qu'il héberge.

En effet, deux applications fonctionnant sur des machines différentes ne génèrent pas de concurrence d'accès aux ressources (entre autres : la mémoire vive et les disques). Ce qui est le cas quand l'ensemble est localisé sur un seul et même serveur.

3. Séparation des applications

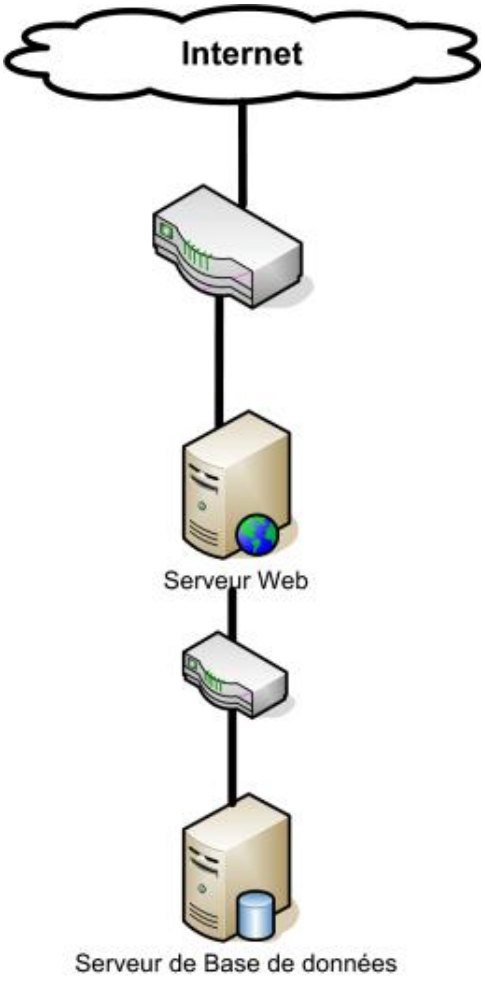
Le premier pas consiste à spécialiser les serveurs. La spécialisation d'un serveur entraîne son paramétrage de manière spécifique à l'hébergement d'un service ou d'une application.

Prenons, par exemple, l'utilisation d'un serveur de base de données. Il devra être installé avec une partition de stockage des fichiers de la base plus importante.

Un serveur Apache devra aussi être paramétré afin de permettre un maximum de parallélisme sur l'accès aux pages Web.

Ainsi, il est plus facile de demander des ressources spécifiques en fonction d'un besoin de montée en charge identifié sur une partie du service.

© ENI Editions - All rigths reserved - Algeria Educ



4. Identification des limites de performance

Les limites de performance d'un serveur sont souvent simples à déterminer :

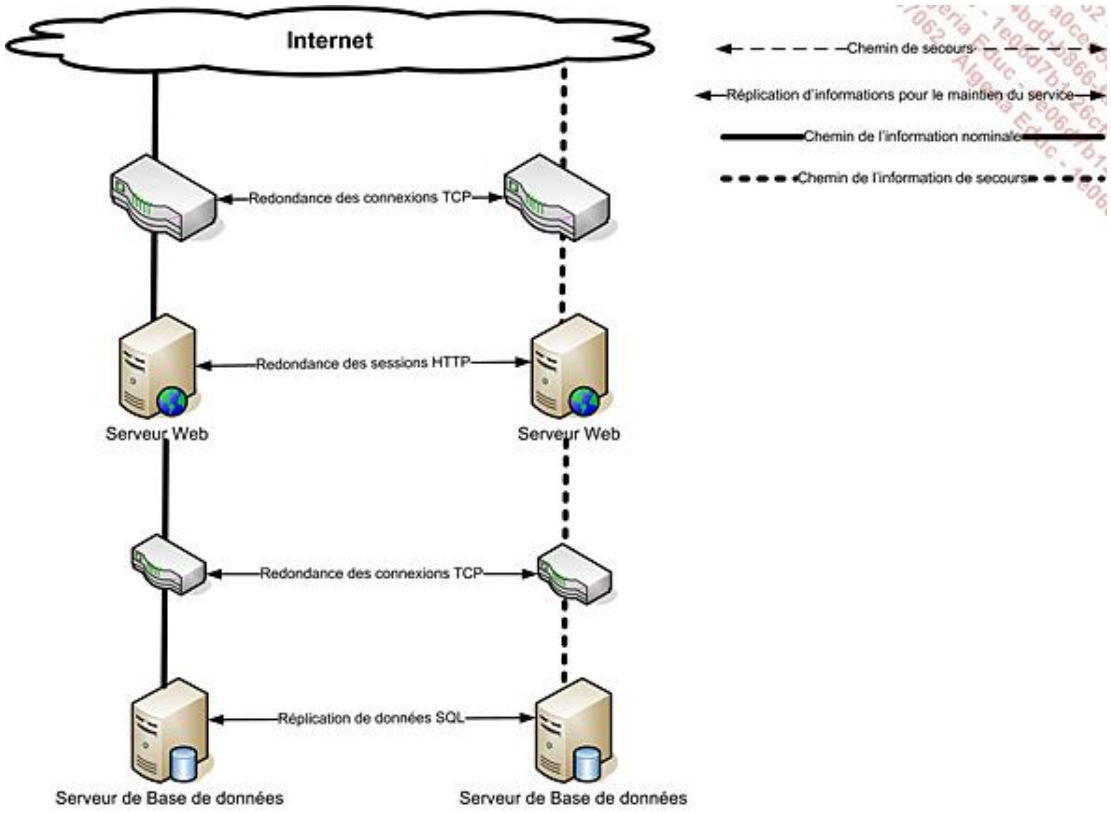
- Mémoire RAM disponible
- Espace disque disponible
- Débit d'entrée du disque
- Débit de sortie du disque
- Charge moyenne du CPU
- Débit d'entrée de la carte réseau
- Débit de sortie de la carte réseau
- Nombre d'entrées RAM
- Nombre de sorties RAM

Avec ces quelques indicateurs, il est beaucoup plus facile de mettre le doigt sur les éléments défaillants d'un système. Cette analyse est facilitée par la spécialisation des serveurs pour un type d'application, limitant aussi l'interaction entre les applications lors de l'accès aux ressources.

© ENI Editions - All rigths reserved - Algeria Educ

5. Haute disponibilité partielle

L'idée est de prévoir un ensemble d'équipements physiques (serveurs, routeurs, pare-feu) identiques n'étant utilisés qu'en cas de problème sur l'architecture normale ou nominale. Il s'agit donc d'une architecture de secours. En cas de problème, le trafic et les appels sont routés vers ces équipements de secours.

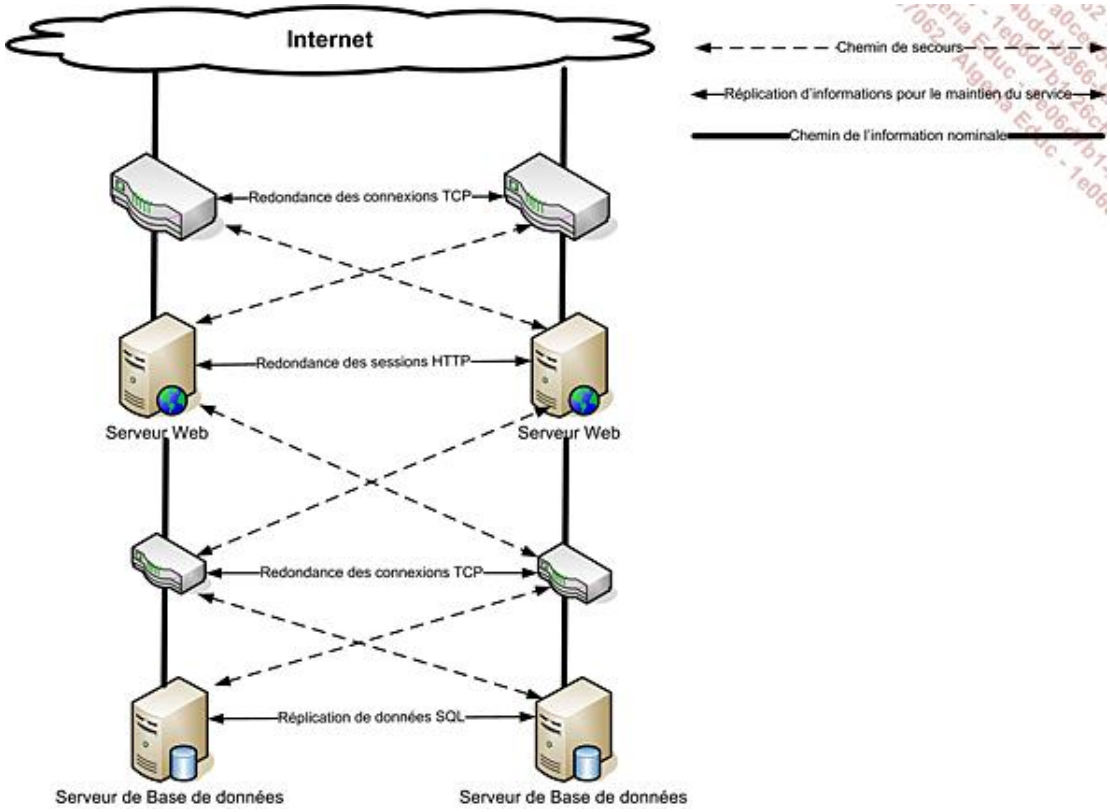


6. Haute disponibilité intégrale

L'idée d'une haute disponibilité de bout en bout conduit forcément à doubler l'ensemble des équipements et des serveurs. Ainsi, le schéma de spécialisation complètement doublé (ou en redondance complète) donne l'architecture système et réseau suivante. Dans l'optique du doublement des équipements, prévoir un système de partage des interrogations entre deux plates-formes complètes semble intéressant car cela permet d'augmenter la capacité de tenue en charge vis-à-vis d'une forte consultation du site.

Dans ce cas, l'ensemble de la plate-forme est en mode actif-actif et l'ensemble des équipements est utilisé pour rendre le service prévu. En cas de défaillance, l'équipement redondant doit être capable d'absorber l'ensemble du trafic et, donc, doit avoir été dimensionné pour cela.

© ENI Editions - All rigths reserved - Algeria Educ



© ENI Editions - All rigths reserved - Algeria Educ

Haute disponibilité des données

Le gros du travail de maintien de la haute disponibilité est la diffusion de l'information dans l'architecture de telle sorte que le service soit toujours rendu aux utilisateurs et que la continuité de service soit opérationnelle. La continuité de service consiste à ne jamais perdre les informations sur les données échangées avec un client afin de maintenir son authentification, ses données temporaires et l'ensemble des informations liées à son activité sur un site.

Pour cela, il est impératif de maintenir cette information en double et à jour entre au moins deux sites.

La réplication des informations concerne tous les niveaux de données de l'architecture :

- Session TCP.
- Session HTTP.
- Fichiers partagés.
- Données de cache.
- Base de données.

Lorsque l'ensemble de ces informations est répliqué, il devient donc possible de garantir à la fois le rendu du service et la continuité de service.

Voyons maintenant en détail chacun des niveaux de réplication ainsi qu'une implémentation possible.

1. Partage de sessions TCP

Les équipements réseau ont la capacité d'implémenter des fonctionnalités de maintien de session TCP, ce qui permet de ne pas avoir à ouvrir une nouvelle connexion TCP. Dans le cas des architectures Web, l'utilisation de ce type de réplication n'est nécessaire que si votre serveur Web est paramétré pour le maintien des connexions, sinon une connexion est recréée à chaque appel HTTP du serveur.

Sous Apache 2.x, pour activer le paramétrage du maintien des connexions TCP, placez la directive suivante à la racine ou dans la configuration de votre hôte virtuel.

```
KeepAlive On
```

Le partage et le maintien des connexions ne sont donc pas absolument nécessaires pour garantir une continuité de service. Leur absence ne provoque pas de coupure ponctuelle pour les utilisateurs.

Cependant, si l'architecture doit supporter des connexions permanentes vers d'autres applications telles que MySQL, le maintien des sessions TCP peut s'avérer primordial, surtout dans le cas où les applications utilisatrices n'ont pas été pensées, conçues ou développées avec des services de gestion des connexions permettant la reprise sur erreur.

2. Haute disponibilité des fichiers

De nombreuses solutions NAS ou SAN existent sur le marché, permettant le partage et la mise en haute disponibilité de disques réseau sur plusieurs machines. Ces serveurs de fichiers réseau gèrent pour vous l'accès concurrent, la sauvegarde à chaud et la redondance des fichiers et des arborescences mises à disposition.

Comme il y a toujours un revers à chaque médaille, l'accès à chaque fichier est plus lent que l'accès à un fichier qui serait disponible sur un disque dur du serveur, une carte flash ou même un fichier en mémoire RAM.

Il est donc primordial de ne partager que les arborescences nécessaires au maintien et à la continuité du service.

Ce type d'arborescence permet de partager des avatars, des enregistrements de transactions ou des fichiers chargés sur le serveur, par exemple.

Il est cependant peu souhaitable de partager le code PHP de votre application, ni la configuration de votre application. En effet, à chaque appel, des échanges entre le serveur Web et le serveur de fichiers seront nécessaires, quel que soit le type des requêtes ou le type d'utilisation.

Ce qu'il faut éviter avant tout, ce sont les échanges de fichiers systématiques à chaque interrogation d'un utilisateur.

© ENI Editions - All rigths reserved - Algeria Educ

3. Haute disponibilité des caches

Le système de cache peut être localisé sur chaque serveur de manière autonome. En effet, en régime nominal, lorsque le service fonctionne correctement, les caches locaux enregistrent les données des utilisateurs qu'ils traitent. La réplication de la donnée de cache est donc un surcoût qui se compense facilement par une nouvelle mise en cache sur un autre serveur.

Il est cependant intéressant de mettre massivement de l'information en cache pour gagner largement en performance.

a. Memcache : le serveur de cache

Memcache se définit comme étant un système de cache distribué d'objets en mémoire, générique par essence et dont le but est de booster la rapidité des sites Web en diminuant la charge des bases de données.

Memcached est un serveur très simple à paramétrer et à utiliser. Il possède de nombreuses API dans la plupart des langages de programmation. Il est utilisé par plus de 85% des 50 sites mondiaux les plus importants en termes de trafic et fait référence.

b. Memcache en haute disponibilité

Il n'est pas conseillé d'utiliser memcache en haute disponibilité pour les raisons suivantes :

- Pas de mécanisme intégré de réplication.
- Pas de stockage fiable.
- Pas d'authentification.
- Pas d'invalidation de données possible.

Il peut être cependant judicieux de localiser les caches sur plusieurs serveurs et de dédier à chacun des serveurs la tâche de stocker spécifiquement un certain type de données, par exemple. L'ensemble ne nécessitera pas de mécanisme de redondance important. Si un serveur tombe, un autre serveur prend le relais par des mécanismes (offerts nativement) de reprise sur incident codés directement en PHP.

La distribution des appels de cache entraîne cependant un « temps de chauffe » pendant lequel les données peuvent être mises plusieurs fois en cache avant une efficacité optimale du système. Le nombre de fois où une donnée peut être mise en cache, dans le pire des cas, est égal au nombre de serveurs introduits dans le pool de connexion Memcache.

c. Connecteur simple PHP à Memcache

L'extension Memcache doit être installée et activée afin de pouvoir utiliser les exemples suivants.

Le principe est simple, on crée un objet Memcache qui va gérer un serveur.

Il va effectuer les actions suivantes :

- Déclaration d'un objet Memcache.
- Ajout d'un serveur Memcache.
- Affichage des statistiques du serveur.
- Envoi de nouvelles entrées dans le cache.
- Récupération des entrées du cache.

© ENI Editions - All rigths reserved - Algeria Educ

- Affichage des statistiques du serveur.

```
<?php
$nb_entrees=10;
$memcache = new Memcache;

$memcache->addServer('localhost', 11211);


print_r($memcache->getExtendedStats());

for ($i=0;$i<$nb_entrees;$i++) {
        $memcache->set("key$i", "Valeur$i");
}

for ($i=0;$i<$nb_entrees;$i++) {
    print "\n\tkey$i =>";
    $val=$memcache->get("key$i");
    if (!$val) echo "NULL";
    else echo "$val";
}
print_r($memcache->getExtendedStats());
?>
```

d. Description d'un cache Web

Il est possible d'ajouter plusieurs serveurs via la méthode addServer de l'objet MemCache.

Dans ce second exemple Code du cache basé sur Memcache, nous pouvons analyser un code PHP générant six serveurs Memcache locaux à la machine écoutant sur les ports compris entre 11211 et 11216.

Cet exemple permet de transformer un serveur Web HTTP en un puissant point d'entrée d'une ferme de serveurs de cache Memcache.

Ce script possède trois modes :

- stat permettant de récupérer des informations statistiques.
- set permettant d'assigner une valeur pour une clé donnée.
- get permettant de récupérer la valeur d'une clé.

Le **mode statistique** permet de récupérer les statistiques brutes de l'ensemble des serveurs. Pour cela, il faut positionner le paramètre action à stat :

http://localhost/memcache/mem.php?action=stat

```
Array
(
    [localhost:11211] => Array
        (
            [pid] => 17293
            [uptime] => 3275
            [time] => 1272555648
            [version] => 1.4.5
            [pointer_size] => 32
            [rusage_user] => 0.048992
            [rusage_system] => 0.052991
            [curr_connections] => 4
            [total_connections] => 6
            [connection_structures] => 5
            [cmd_get] => 1718
            [cmd_set] => 1718
            [cmd_flush] => 0
            [get_hits] => 1718
            [get_misses] => 0
            [delete_misses] => 0
```

© ENI Editions - All rigths reserved - Algeria Educ

```
            [delete_hits] => 0
            [incr_misses] => 0
            [incr_hits] => 0
            [decr_misses] => 0
            [decr_hits] => 0
            [cas_misses] => 0
            [cas_hits] => 0
            [cas_badval] => 0
            [auth_cmds] => 0
            [auth_errors] => 0
            [bytes_read] => 76521
            [bytes_written] => 81524
            [limit_maxbytes] => 67108864
            [accepting_conns] => 1
            [listen_disabled_num] => 0
            [threads] => 4
            [conn_yields] => 0
            [bytes] => 114427
            [curr_items] => 1717
            [total_items] => 1718
            [evictions] => 0
            [reclaimed] => 0
        )

...

)
```

Le mode **ajout de cache** est très simple et renvoie 1 si l'opération s'est bien passée, 0 sinon :

http://localhost/memcache/mem.php?action=set&k=cle1&v=mavaleurcle1

Il suffit de positionner le paramètre action à set puis d'indiquer la valeur de la clé au travers du paramètre k et le contenu ou la valeur au travers du paramètre v.

Le mode **interrogation du cache** est très simple et renvoie vide si rien n'est trouvé ou bien le contenu du cache pour la clé spécifiée.

http://localhost/memcache/mem.php?action=get&k=cle1

```
mavaleurcle1
```

http://localhost/memcache/mem.php?action=get&k=cle134

L'appel à l'URL ci-dessus renvoie une chaîne vide.

Il suffit de positionner le paramètre action à set puis d'indiquer la valeur de la clé au travers du paramètre k et le contenu ou la valeur au travers du paramètre v.

e. Code du cache basé sur Memcache

```
<?php
$nb_serveurs=6;
$nb_entrees=10000;
$memcache = new Memcache;
for ($i=1;$i<=$nb_serveurs;$i++) {
    $memcache->addServer('localhost', 11210+$i);
}

if ( $_GET['action']=='stat') {
    echo "<h1>Statistiques:</h1><pre>";
        print_r($memcache->getExtendedStats());
    echo "</pre>";
    exit(0);
}

if ( $_GET['action']=='set') {
    $st=$memcache->set($_GET['k'], $_GET['v']);
    if (!$st) $st=$memcache->replace($_GET['k'], $_GET['v']);
    echo $st;
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    exit(0);
}

if ( $_GET['action']=='get') {
    print $memcache->get($_GET['k']);
}


if ( $_GET['action']=='test') {
    echo "<h1>Statistiques avant:</h1><pre>";
    print_r($memcache->getExtendedStats());
    echo "</pre>";

    for ($i=0;$i<$nb_entrees;$i++) {
        $memcache->set("key$i", "Valeur$i");
    }

    echo "<h1>Statistiques après ajout:</h1><pre>";
    print_r($memcache->getExtendedStats());
    echo "</pre>";

    for ($i=0;$i<$nb_entrees;$i++) {
        print "<li>Key $i :";
        $val=$memcache->get("key$i");
        if (!$val) echo "NULL";
        else echo "$val";
        print "</li>";
    }

    echo "<h1>Statistiques après lecture:</h1><pre>";
    print_r($memcache->getExtendedStats());
    echo "</pre>";
}

?>
```

f. Lancement des serveurs Memcached sous Linux

Pour la gestion du lancement local des serveurs Memcached, nous utilisons sous Linux le programme binaire provenant directement du site Web.

http://www.memcached.org

L'exécution des serveurs s'appuie sur trois paramètres majeurs.

- L'option -p permet de définir le port TCP d'écoute du serveur. En effet, il n'est pas possible de lancer plusieurs Memcached sur le même port TCP.
- L'option -d permet de lancer le programme en tant que démon UNIX ou service Windows.
- L'option -P permet de définir le fichier contenant l'identifiant du processus du serveur. Ce fichier, dans notre exemple, permettra l'arrêt des serveurs et la consultation du statut d'exécution de chacun des serveurs.

```
#!/bin/sh

ROOT=/home/jmrenouard/memcached-1.4.5
NB_INSTANCE=6
function start {
        echo "* Lancement des serveurs"

        for i in `seq 1 $NB_INSTANCE`; do
                echo -e "\tLancement instance $i sur le port
1121$i"
                $ROOT/memcached -U0 -p 1121$i -d -P /tmp/memcache.$i
        done
}
```

© ENI Editions - All rigths reserved - Algeria Educ

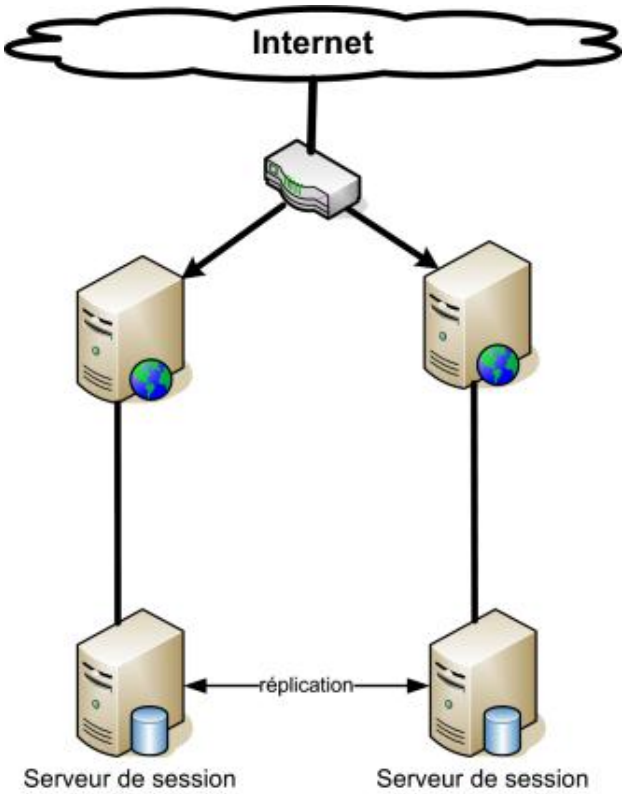
```
function stop {
        echo "* Arret des serveurs"
        for i in `seq 1 $NB_INSTANCE`; do
                echo -e "\tArret instance $i sur le port 1121$i"
                kill -9 `cat /tmp/memcache.$i`
        done
}

function status {
        echo "* Status des serveurs"
        for i in `seq 1 $NB_INSTANCE`; do
                echo -en "\tEtat instance $i sur le port 1121$i => "
                ps -edf | grep `cat /tmp/memcache.$i`| grep -v
grep || echo
        done

}


CMD=${1:-"status"}

$CMD
```

4. Haute disponibilité des bases de données

La haute disponibilité d'une base de données passe par l'utilisation des mécanismes de réplication de manière circulaire, tout en évitant les boucles infinies de copie des données.

Pour cela, voici une architecture type avec un socle de deux bases de données MySQL montées en maître/maître afin de garantir la disponibilité des données ainsi que le partage des connexions entre deux serveurs de base de données.

5. Architecture type



Dans notre exemple, nous avons décidé de nous appuyer sur les mécanismes de réplication de MySQL mis en œuvre

© ENI Editions - All rigths reserved - Algeria Educ

en configuration maître/maître. Cette configuration permet à deux serveurs de partager les ajouts, les modifications et les suppressions réalisés par un serveur en les réexécutant sur l'autre serveur de base de données. Le premier serveur écoute les modifications du second et le second écoute les modifications du premier. La configuration permet cependant de ne pas propager les modifications de manière circulaire entre les deux serveurs.

a. Configuration MySQL du premier serveur

```
...
report-host=serveur1
log-bin=/var/lib/mysql/mysql-bin
server-id=1
auto_increment_increment=2
auto_increment_offset=1
relay-log=relay-bin
log-slave-updates
replicate-same-server-id=0
...
```

b. Configuration MySQL du deuxième serveur

```
...
report-host=serveur2
log-bin=/var/lib/mysql/mysql-bin
server-id=2
auto_increment_increment=2
auto_increment_offset=2
relay-log=relay-bin
log-slave-updates
replicate-same-server-id=0
...
```

c. Détails de la configuration

La réplication s'appuie sur les traces binaires (ou binary log). Chaque requête de modification est stockée dans les traces binaires et consommée par l'autre serveur via le mécanisme de réplication natif à MySQL. Le paramètre est `log-bin` et il déclenche le stockage des traces binaires. Il est possible de donner le nom du fichier de manière absolue ou relative.

Le paramètre `report-host` permet de donner la possibilité au serveur maître de voir la liste des esclaves connectés.

Pour faciliter la propagation des modifications au plus vite, il est possible de paramétrer un fichier de relais, de stocker localement les modifications du serveur maître et de consommer les modifications distantes depuis des données locales. Le paramètre qui active localement la copie des modifications est `relay-log`.

Le but est de permettre la réplication entre serveurs sans entraîner d'erreur dans la création des identifiants de base de données. Pour cela, il suffit d'imposer à l'un des serveurs de ne générer que des identifiants de clé pairs et de demander à l'autre serveur de ne générer que des identifiants impairs. De ce fait, il est impossible que les deux serveurs génèrent au même moment deux identifiants identiques. Lorsque deux serveurs génèrent le même identifiant au même moment, la réplication des données est stoppée jusqu'à résolution du conflit sur chacun des serveurs. Le premier paramètre à utiliser est `auto_increment_increment`, qui indique le pas entre deux identifiants. Ici le pas est de deux ce qui signifie que quels que soient deux identifiants dans une table donnée, ils ne peuvent pas avoir une différence absolue inférieure à 2.

Le deuxième paramètre `auto_increment_offset` indique le point de départ des identifiants de table.

Ces deux paramètres contrôlent plus largement l'ensemble du fonctionnement des colonnes ayant comme valeur AUTO_INCREMENT. Le paramètre AUTO_INCREMENT donne la permission à MySQL de générer lui-même les identifiants.

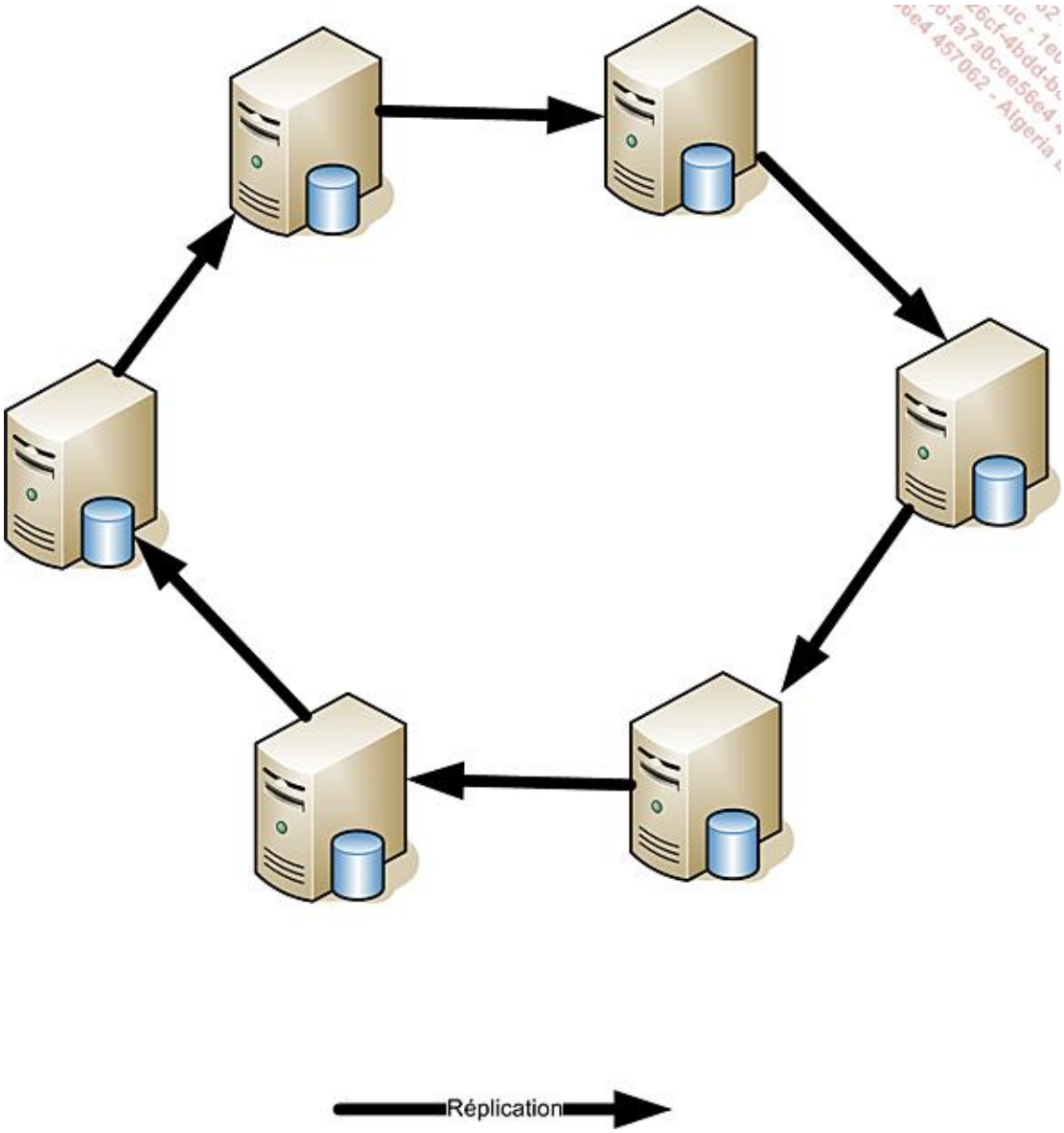
d. Limites de la configuration maître/maître

La limite de ce type de configuration reste essentiellement le nombre de maîtres qu'il est préférable de limiter à deux. En effet, multiplier le nombre de maîtres dans la boucle des maîtres vient donc à augmenter la probabilité d'erreur et de points de faiblesse dans votre architecture.

Cette configuration, de plus, nécessite des paramétrages et du code spécifique capable de récupérer les identifiants

© ENI Editions - All rigths reserved - Algeria Educ

des éléments insérés en table plutôt que de les fabriquer directement.



Cette illustration avec six serveurs montre que l'ensemble des maîtres nécessite le fonctionnement de six réplications simultanées. De plus, chaque modification de données nécessite cinq réplications pour mettre l'ensemble des serveurs à niveau.

Il est donc impératif de définir un serveur maître de référence ainsi que la chaîne complète de remise à niveau des serveurs en cas de problème sévère sur l'anneau de réplication.

e. Solutions de contournement des limites

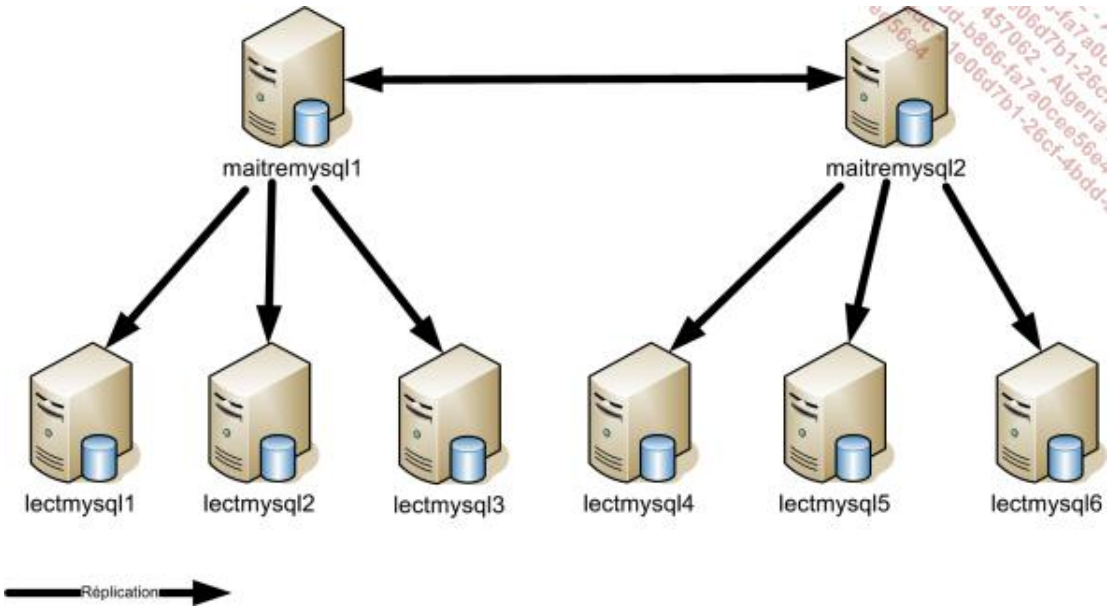
Afin de pallier la complexité des anneaux de maîtres et permettre une montée en charge des serveurs, il peut être intéressant d'ajouter un niveau de hiérarchie afin de permettre l'établissement d'un ensemble de serveurs esclaves pour la consultation en lecture uniquement. Cette approche permet d'offrir un point d'entrée pour les modifications des données et un autre point d'entrée pour les consultations en lecture.

Afin de permettre cela par défaut, il faut absolument valider la présence des deux lignes suivantes dans votre configuration de MySQL.

```
log-slave-updates
replicate-same-server-id=0
```

log-slave-updates indique que toutes les données de modification doivent être transmises à l'ensemble de ses esclaves permettant la réplication des données provenant d'un autre serveur. Cependant, afin de ne pas engendrer une boucle de commande de réplication, l'option replicate-same-server-id=0 interdit la transmission des modifications venant de lui-même.

© ENI Editions - All rigths reserved - Algeria Educ



Il est donc possible de construire un second niveau de paramétrage.

En vue d'une optimisation, il faut trouver une solution permettant de transmettre l'ensemble des lectures (SELECT) à l'un des serveurs esclaves et toute modification (DELETE, UPDATE, INSERT) vers l'un des deux maîtres.

Pour cela, il est possible d'utiliser une solution telle que Mysql-Proxy permettant de définir des serveurs pour la lecture uniquement. Mysql-Proxy est disponible sous licence GPL version 2 à l'adresse suivante : http://forge.mysql.com/wiki/MySQL_Proxy ; il permet, via le langage Lua, de configurer finement le comportement de votre relais de requêtes SQL.

```
[mysql-proxy]
proxy-read-only-backend-addresses = lectmysql1:3306,
lectmysql2:3306, lectmysql3:3306, lectmysql4:3306,
lectmysql5:3306,lectmysql6:3306
proxy-backend-addresses = maitremysql1:3306,
maitremysql2:3306
```

Ce programme possède toutes les fonctionnalités permettant le transfert de requêtes SQL selon leur nature. De base, leur fonctionnement envoie les lectures sur l'ensemble des serveurs. Seuls les serveurs « `proxy-read-only-backend-addresses` » recevront des requêtes de lecture (SELECT).

Cependant, il est à noter que de nombreuses zones d'ombre persistent sur l'utilisation en production de Mysql-Proxy car de nombreux points concernant la récupération du dernier identifiant inséré et de la récupération du nombre de lignes récupérées ne donnent pas les bons résultats. En effet, les interrogations ne partent pas vers le même serveur à chaque requête. Ceci empêche tout simplement l'utilisation de réplication maître/maître.

La seconde option, plus complexe car plus spécifique est d'inclure directement dans le code des connecteurs PDO distincts qui seront sélectionnés en fonction de la nature des requêtes. Cette approche est moins générique et pose des problèmes liés à la qualité. En effet, il faut un temps certain avant que ne se produisent la validation et la correction de l'ensemble des bugs, donc prudence.

Une troisième alternative consiste à inclure une classe générique émulant la couche PDO de PHP par extension. Cette couche aura sa classe de paramétrage propre et permettra de diriger les appels vers les bons serveurs par défaut.

Comme nous l'avons vu, il n'existe pas aujourd'hui de solution de relai générique pour MySQL permettant de répartir le trafic de manière correcte en environnement de production sans passer par l'écriture de sa propre solution logicielle.

© ENI Editions - All rigths reserved - Algeria Educ

Présentation des sessions HTTP

Dans le protocole HTTP, la session HTTP est le lien unique entre le visiteur unique et le serveur, permettant au serveur de maintenir des informations entre différents appels d'un utilisateur, sans conflit entre les différents accès simultanés des autres utilisateurs.

Le protocole HTTP est un protocole sans sauvegarde de l'état entre deux appels et donc la session HTTP est un moyen d'ajouter cette fonctionnalité à vos applications PHP. Chaque session est caractérisée par un numéro de session unique : l'identifiant de session ou sessionid. La session est maintenue par l'envoi d'un cookie à la première connexion, ce cookie a généralement une durée de vie très courte. Ce cookie de session est le lien côté client permettant de retrouver les informations sauvegardées sur le serveur.

1. Stockage des informations en PHP

Les sessions en PHP sont stockées sur un espace disque dont le chemin est indiqué par le paramètre de php.ini.

Paramètre	Description
session.save_handler	Identifiant du mécanisme de sauvegarde de la session
session.save_path	Paramètre de connexion ou chemin d'accès à la ressource

Il existe deux grands moyens standard de stocker les sessions :

- File : utilisation du disque pour stocker les informations de session.
- Memcache : utilisation d'un serveur Memcached pour le partage des sessions.

Il est possible d'indiquer le serveur Memcached par paramétrage pour y stocker les informations de sessions.

```
...
session.save_handler = memcache
session.save_path = "tcp://localhost :11211"
...
```

Il est plus simple d'utiliser le mécanisme à base de fichiers par défaut.

```
...
session.save_handler = files
session.save_path = "/var/lib/php/session"
...
```

2. Redéfinition de la gestion de session

Pour ne pas perdre l'information des sessions et garantir une continuité totale et transparente d'un service et ceci malgré une défaillance matérielle, il est impératif de doubler les serveurs Web traitant les requêtes HTTP des clients et aussi les équipements de stockage des sessions.

Il n'existe pas de moyen natif dans les serveurs Web actuels permettant le partage des sessions. Ce type de mécanisme est souvent complexe à gérer et à paramétrer et n'est disponible qu'avec des serveurs d'application.

Les serveurs Web de consultation sont interrogés directement et reçoivent le trafic et les interrogations des utilisateurs. Leur rôle est de bien gérer les interrogations et fournir la meilleure réponse possible à chaque utilisateur en un temps acceptable.

Le mécanisme de sauvegarde des réplications peut s'appuyer soit sur un cache distribué, soit sur une base de données répliquée entre les deux serveurs. Cependant, les données du cache ou de la base doivent être distribuées afin de pallier les défaillances techniques des serveurs hébergeant le système de cache.

a. Classes PEAR de gestion des sessions partagées

© ENI Editions - All rigths reserved - Algeria Educ

L'installation se passe en plusieurs phases :

- Installation de la classe DB.
- Mise à jour du channel PEAR.
- Installation de la version beta de http_Session2.
- Installation de la classe MDB2.
- Installation du pilote Mysql pour la classe MDB2.

Il est à noter que la classe de gestion Memcache est native à PHP et permet la gestion directe des accès aux serveurs de cache Memcache.

```
# pear install DB
WARNING: "pear/DB" is deprecated in favor of "pear/MDB2"
WARNING: channel "pear.php.net" has updated its protocols, use
"pear channel-update pear.php.net" to update
downloading DB-1.7.13.tgz ...
Starting to download DB-1.7.13.tgz (132,246 bytes)
.............................done: 132,246 bytes
install ok: channel://pear.php.net/DB-1.7.13

# pear channel-update pear.php.net
Updating channel "pear.php.net"
Update of Channel "pear.php.net" succeeded

# pear install Http_Session2
Failed to download pear/Http_Session2 within preferred state
"stable", latest re
lease is version 0.7.3, stability "beta", use
"channel://pear.php.net/Http_Sessi
on2-0.7.3" to install
install failed

C:\wamp\bin\php\php5.2.11>pear install
channel://pear.php.net/HTTP_Session2-0.7.3
WARNING: "pear/DB" is deprecated in favor of "pear/MDB2"
Did not download optional dependencies: pear/MDB2, use --alldeps
to download automatically
pear/HTTP_Session2 can optionally use package "pear/MDB2" (version
>= 2.4.1)
downloading HTTP_Session2-0.7.3.tgz ...
Starting to download HTTP_Session2-0.7.3.tgz (16,362 bytes)
......done: 16,362 bytes
install ok: channel://pear.php.net/HTTP_Session2-0.7.3

# pear install MDB2
downloading MDB2-2.4.1.tgz ...
Starting to download MDB2-2.4.1.tgz (119,790 bytes)
...........................done: 119,790 bytes
install ok: channel://pear.php.net/MDB2-2.4.1
MDB2: Optional feature fbsql available (Frontbase SQL driver for
MDB2)
MDB2: Optional feature ibase available (Interbase/Firebird driver
for MDB2)
MDB2: Optional feature mysql available (MySQL driver for MDB2)
MDB2: Optional feature mysqli available (MySQLi driver for MDB2)
MDB2: Optional feature mssql available (MS SQL Server driver for
MDB2)
MDB2: Optional feature oci8 available (Oracle driver for MDB2)
MDB2: Optional feature pgsql available (PostgreSQL driver for
MDB2)
MDB2: Optional feature querysim available (Querysim driver for
MDB2)
```


© ENI Editions - All rigths reserved - Algeria Educ

```
MDB2: Optional feature sqlite available (SQLite2 driver for MDB2)
MDB2: To install optional features use "pear install
pear/MDB2#featurename"

# pear install MDB2#mysql
Skipping package "pear/MDB2", already installed as version 2.4.1
downloading MDB2_Driver_mysql-1.4.1.tgz ...
Starting to download MDB2_Driver_mysql-1.4.1.tgz (36,481 bytes)
..........done: 36,481 bytes
install ok: channel://pear.php.net/MDB2_Driver_mysql-1.4.1
```

b. Code de gestion des sessions en base via PEAR

Notre exemple nous montre comment utiliser un script PHP afin de stocker les sessions dans une table de base de données. Pour cela nous nous appuierons sur la classe http_Session2 et son extension : MDB2 pour MySQL. Il est important de noter que la classe MDB2 provoque des alertes d'utilisation dépréciées qui viendront à disparaître dans les futures versions de PHP.

Il est donc à noter que ce code est relativement simple et facile à comprendre.

Code de création SQL de la table de stockage des sessions

```
CREATE TABLE SESSION (
    id     varchar(32) NOT NULL,
    expiry int(10),
    data   text,
    PRIMARY KEY (id)
);
```

Exemple d'utilisation de session en base de données

```
<?php

require_once 'HTTP/Session2.php';
ini_set ('error_reporting', E_ALL&~ E_DEPRECATED);
HTTP_Session2::useTransSID(false);
HTTP_Session2::useCookies(false);

HTTP_Session2::setContainer('MDB2',
    array('dsn' => 'mysql://session:session@localhost/session',
        'table' => 'SESSION'));

#
HTTP_Session2::useCookies(true);
HTTP_Session2::start('compteur');
HTTP_Session2::setExpire(time() + 60); //  60 secondes de durée de
vie
HTTP_Session2::setIdle(time() + 5);    // 5 secondes d'inactivité
if (HTTP_Session2::isExpired()) {
    HTTP_Session2::destroy();
}

if (HTTP_Session2::isIdle()) {
    echo "<br/>Session inactive.";
    HTTP_Session2::destroy();
}
if (HTTP_Session2::isNew()) {
      echo 'NOUVELLE SESSION';
    HTTP_Session2::set('nbVisite', "0") ;
   } else {
   $nbv=HTTP_Session2::get('nbVisite')+1;
   HTTP_Session2::set('nbVisite', "$nbv") ;

    echo 'RECUPERATION SESSION => ' . $nbv.  ' Visite(s).'     ;
}
```

© ENI Editions - All rigths reserved - Algeria Educ

```
 HTTP_Session2::updateIdle();
?>
```

Résultat d'utilisation de la première fois

NOUVELLE SESSION

Résultat d'utilisation de la seconde fois

RECUPERATION SESSION => 1 Visite(s).

c. Code de gestion des sessions en base sans PEAR

Notre exemple nous montre comment utiliser un script PHP afin de stocker les sessions dans une table de base de données. Pour cela nous nous appuierons sur l'API MySQL de base fournie par PHP, et rien d'autre.

Code SQL de génération de la table

```
  create table SESSION_SANS_PEAR(
  id_table int not null auto_increment,
  id text not null,
  data text,
  expire int not null,
  primary key(id_table)
  );
```

Code complet sans utilisation de PEAR

```
<?php

// config
$host = "localhost";
$user = "session";
$pass = "session";
$dbname = "session";
$table = "SESSION_SANS_PEAR";

/* fonction de création de session*/
function open ($s,$n)
{
    global $session_connection, $host, $user, $pass, $dbname;

    $session_connection = mysql_pconnect($host, $user, $pass);
    mysql_select_db($dbname, $session_connection);
    echo "<br/>Ouverture connexion Mysql";
    return true;
}

/* fonction de lecture de la session */
function read ($id)
{
    global $session_connection,$session_read;
    $query = "SELECT data FROM SESSION_SANS_PEAR WHERE
id=\"{$id}\"";
    $res = mysql_query($query,$session_connection);

    $ret="";
    if(mysql_num_rows($res) == 1) {
        $session_read = mysql_fetch_assoc($res);
        $session_read["data"] =
base64_decode($session_read["data"]);
        $ret=$session_read["data"];
    }
    echo "<br/>Lecture session : $id";
    print "<pre>";print_r($ret);print "</pre>";
```


© ENI Editions - All rigths reserved - Algeria Educ

```
    return $ret;
}

/* fonction d’écriture de la session*/
function write ($id,$data)
{
    echo "<br/>Ecriture session : $id";
    print "<pre>";print_r($data);print "</pre>";

    if(!$data) { return false; }
    echo "<br/>Data non null : $id";

    global $session_connection, $session_read, $session_expire;

    $expire = time() + $session_expire;
    $data = mysql_real_escape_string(base64_encode($data));

    if($session_read) $query = "UPDATE SESSION_SANS_PEAR SET
data=\"{$data}\", expire=\"{$expire}\" WHERE id=\"{$id}\"";
    else $query = "INSERT INTO SESSION_SANS_PEAR SET id=\"{$id}\",
data=\"{$data}\", expire=\"{$expire}\"";

    print "<pre>SQL: $query</pre>";
    mysql_query($query,$session_connection);
    print "<br/>".mysql_errno($session_connection) . ": " .
mysql_error($session_connection) ;
echo "<br/>Ecriture session : OK";
    return true;
}

/* fermeture de la session*/
function close () {
    global $session_connection;
    mysql_close($session_connection);
    echo "<br/>Fermeture session";
    return true;
}

/* suppression de la session */
function destroy ($id) // do not modify function parameters
{
    global $session_connection,$table;
    $query = "DELETE FROM SESSION_SANS_PEAR WHERE id=\"$id\"";
    mysql_query($query,$session_connection);
    echo "<br/>Destruction session : $id";

    return true;
}

/* fonction de suppression des anciennes sessions */
function gc ($expire)
{
    global $session_connection,$table;
    $query = "DELETE FROM SESSION_SANS_PEAR WHERE expire <
".time();
    mysql_query($query,$session_connection);
    echo "<br/>Nettoyage session";

}

/* Positionnement des fonctions de gestion des sessions */
session_set_save_handler ("open", "close", "read", "write",
"destroy", "gc");

/* démarrage de la session */
session_start();

?>
```

© ENI Editions - All rigths reserved - Algeria Educ

Exemple d'application

```
<?php

require_once 'session.php';

if (!isset($_SESSION['nbVisite'])) {
    echo 'NOUVELLE SESSION';
    $_SESSION['nbVisite']=0;
} else {
    $_SESSION['nbVisite']++;
   echo 'RECUPERATION SESSION => ' . $_SESSION['nbVisite'].  '
Visite(s).';
}

?>
```

d. Code de gestion des sessions via Memcache

Notre exemple nous montre comment utiliser un script PHP afin de stocker les sessions dans un serveur de cache Memcached. L'idée est que le serveur de cache applique un algorithme LRU. L'élément le moins utilisé est supprimé du serveur. Pour la gestion des sessions, si la taille du cache est correctement dimensionnée il n'y a pas de meilleure solution d'autorégulation du stockage des données de session. En effet, les sessions non utilisées seront toujours supprimées dans le temps car elles ne seront plus utilisées.

Il est donc à noter que ce code est relativement simple et facile à comprendre. Le serveur Memcached se trouve sur le serveur cache1.server

Exemple d'utilisation de session en serveur de cache

```
<?php

require_once 'HTTP/Session2.php';
ini_set ('error_reporting', E_ALL&~ E_DEPRECATED);
HTTP_Session2::useTransSID(false);
HTTP_Session2::useCookies(false);

$memcache = new Memcache;
$memcache->connect('memcache_host', 11211);

HTTP_Session2::setContainer('Memcache',
    array('memcache' => $memcache,
        'prefix' => 'SESSION_'));

#
HTTP_Session2::useCookies(true);
HTTP_Session2::start('compteur');
HTTP_Session2::setExpire(time() + 60); //  60 secondes de durée de
vie
HTTP_Session2::setIdle(time() + 5);    // 5 secondes d'inactivité
if (HTTP_Session2::isExpired()) {
    HTTP_Session2::destroy();
}

if (HTTP_Session2::isIdle()) {
    echo "<br/>Session inactive.";
    HTTP_Session2::destroy();
}
if (HTTP_Session2::isNew()) {
      echo 'NOUVELLE SESSION';
    HTTP_Session2::set('nbVisite', "0") ;
   } else {
   $nbv=HTTP_Session2::get('nbVisite')+1;
   HTTP_Session2::set('nbVisite', "$nbv") ;

   echo 'RECUPERATION SESSION => ' . $nbv.  ' Visite(s).'    ;
}
```


© ENI Editions - All rigths reserved - Algeria Educ

```
 HTTP_Session2::updateIdle();
?>
```

Avec trois solutions de base pour stocker et partager une session entre plusieurs serveurs Apache et PHP, la continuité de service peut être facilement implémentée et servir de base pour un rendu de qualité.

D’expérience, ce type d’approche offre aussi une sérénité importante lors de la mise en production d’une nouvelle version applicative car lorsque vous installerez une partie de l’application vous serez certain que cela n’aura aucun impact sur le rendu de service vis-à-vis des utilisateurs.

© ENI Editions - All rigths reserved - Algeria Educ

Introduction

Ce chapitre présente la notion d'environnement et la manière de la définir de façon efficace pour vos besoins selon les standards les plus courants. Puis l'ensemble des aspects concernant la mise en place et le paramétrage applicatif d'un environnement de production stable et administrable.

© ENI Editions - All rigths reserved - Algeria Educ

Définition des environnements

Afin de gérer au mieux vos applications et leur cycle de vie, il est impératif de mettre en place plusieurs environnements étanches, c'est-à-dire n'ayant aucun contact ni impact entre eux. Il est important de définir deux points importants :

- Les environnements utilisés.
- Le processus de gestion du cycle de l'application.

Il est important avant toute mise en place d'équipe d'exploitation, de prévoir comment et par quel processus l'application va être déployée en production de la manière la plus fiable et la plus sécurisée possible.

Maîtriser le passage d'un environnement à l'autre devient un point important, permettant d'automatiser intégralement la manière dont les applications sont installées, et ceci quelle que soit la plate-forme.

Maîtriser le processus de déploiement est impératif quelle que soit la technologie utilisée !

Afin de mieux comprendre ce qui se cache derrière la notion d'environnement, voici la présentation des environnements dits standards permettant la gestion optimale des applications.

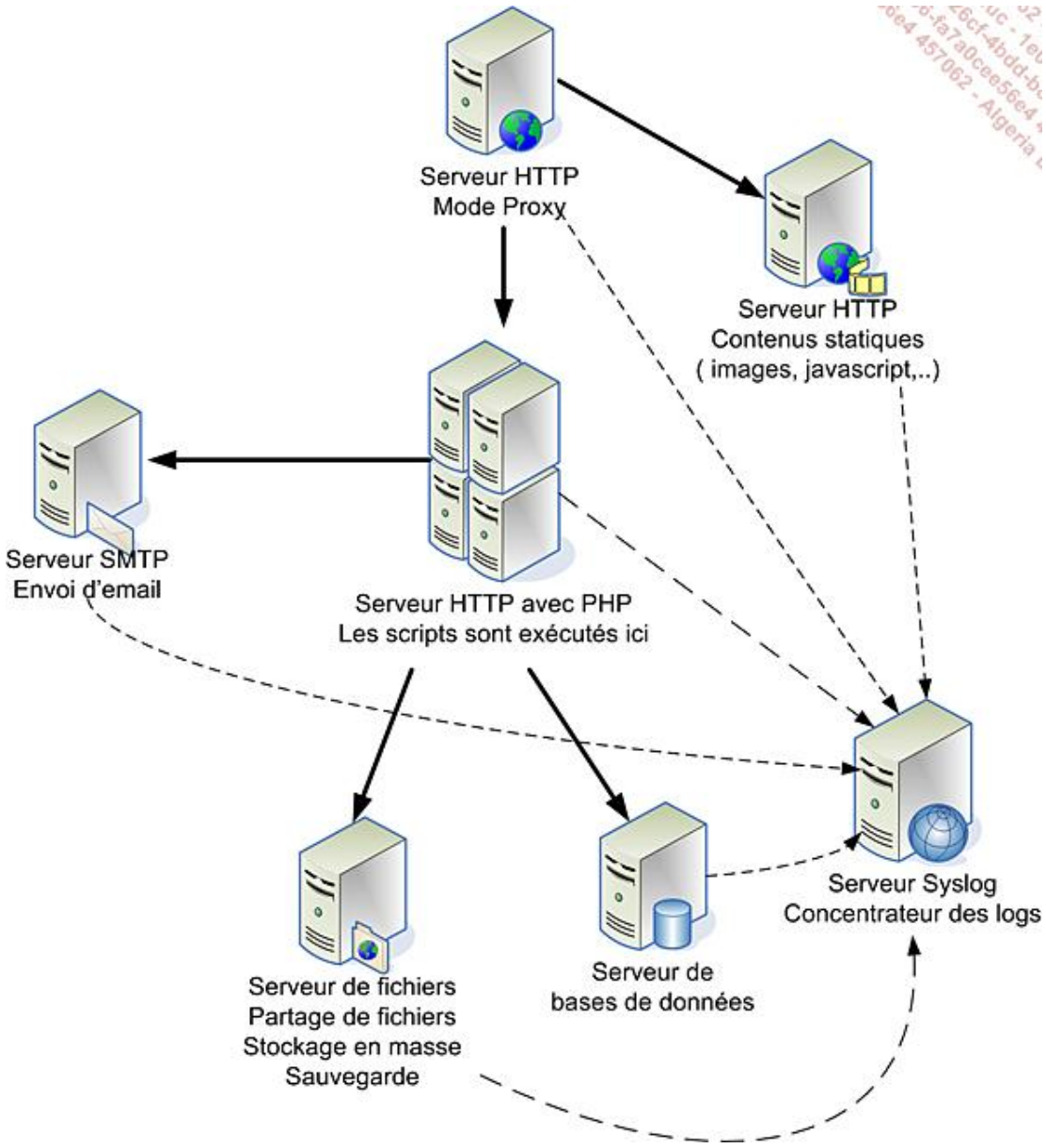
1. Caractéristiques des environnements

Un environnement est un ensemble de ressources matérielles et logicielles permettant le fonctionnement et l'évolution d'une application logicielle. Cette application logicielle est la partie la plus spécifique du produit. Il s'agit souvent d'un site Web ou d'un service Web ayant une apparence et des fonctionnalités mettant en avant une marque ou une société.

Un environnement est un ensemble de ressources autonomes des autres environnements, c'est-à-dire qu'il ne dépend pas d'un autre ensemble de ressources pour fonctionner ni garantir son étanchéité et son indépendance.

Un environnement contient donc l'ensemble des solutions techniques afin de garantir le minimum d'impact entre environnements.

© ENI Editions - All rigths reserved - Algeria Educ



Les ressources les plus communes dans un environnement Web et PHP sont les suivantes :

- Serveur Apache
- Module PHP
- Serveur MySQL
- Serveur Syslog
- Serveur de fichiers partagés type NFS
- Client Subversion
- Client et module d'envoi de Mail

Les ressources partagées entre les environnements dans un contexte Web et PHP sont les suivantes :

- Serveur de gestion de source : Subversion, CVS, CodeSafe, GIT, etc.

© ENI Editions - All rigths reserved - Algeria Educ

- Serveur SMTP tel que Postfix ou Sendmail
- Serveur de partage de fichiers type FTP

Un environnement se caractérise donc par son étanchéité et sa non-intrusion sur d'autres environnements.

2. Environnement de développement

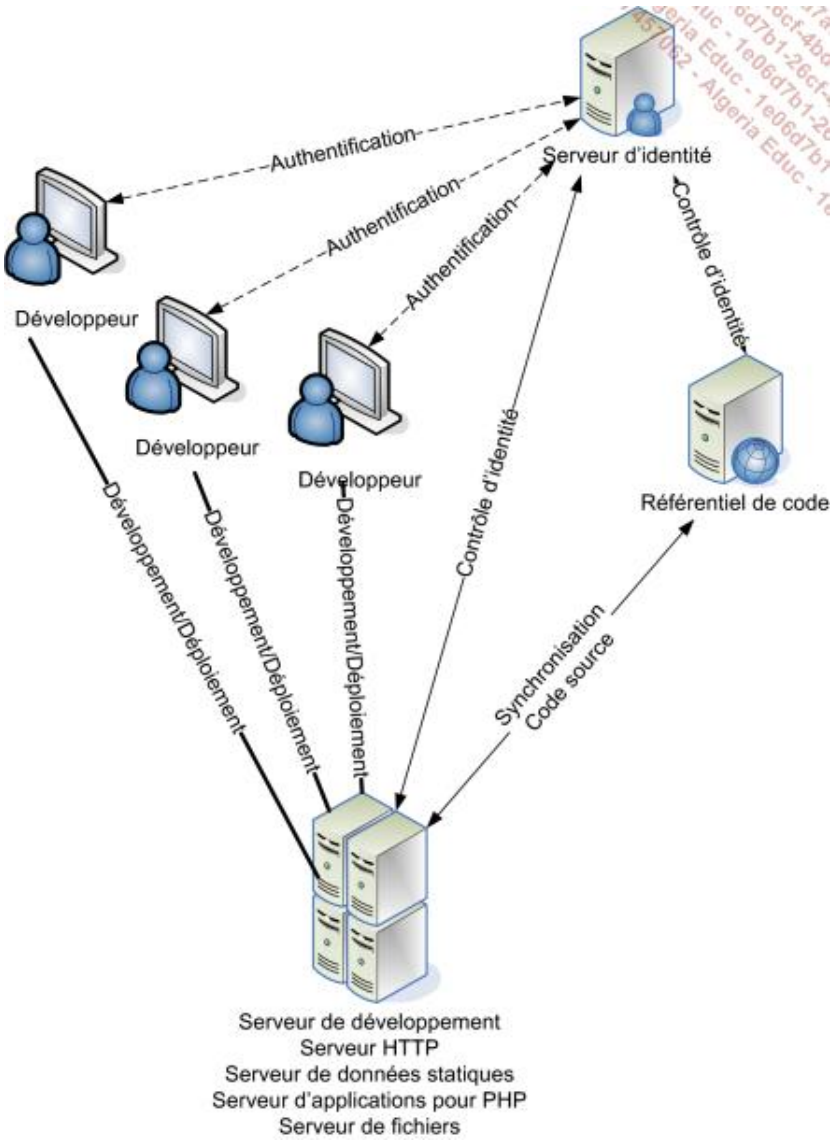
L'environnement de développement est un des domaines fondamentaux dans le cycle de développement permettant à un développeur ou à une équipe de développement de réaliser des évolutions et des corrections du code source tout en garantissant tout en garantissant le fait que les solutions déjà déployées ne soient pas impactées ni sollicitées par l'évolution du code PHP. Cet environnement est généralement limité en accès à un nombre restreint de personnes. Il est constitué de l'ensemble des briques logicielles et des serveurs nécessaires pour un fonctionnement autonome de l'environnement.

Cet environnement est caractérisé par :

- L'instabilité des fonctionnalités due à l'évolution rapide du code.
- L'indisponibilité régulière due aux corrections et aux changements du code.
- L'instabilité des versions : le but des développeurs est la création de la future version.

Les sources sont contrôlées et marquées par lot, comme des versions, permettant leur identification pour d'autres environnements. Pour cela rien n'est plus important pour gérer le développement collaboratif que d'utiliser un gestionnaire de sources tel que CVS, subversion ou bien d'autres.

© ENI Editions - All rigths reserved - Algeria Educ



L'environnement de développement s'appuie sur quatre pivots majeurs :

a. La gestion des droits et des autorisations

Le but est de limiter les droits de lecture et d'écriture à un ensemble restreint de ressources afin de n'offrir que le nécessaire et de sécuriser l'accès à la totalité du code source des applications qui constitue actuellement le plus grand des patrimoines de l'entreprise : le patrimoine informationnel.

b. La centralisation du code source

Au travers du référentiel unique de gestion des sources, il est possible de garantir une cohérence entre tous les développements réalisés. Considéré comme la seule référence fiable, il permet de gérer de larges équipes de développeurs et de garantir l'unicité du code utilisé. Il permet aussi une détection et une résolution rapide des conflits.

De plus, le référentiel unique de gestion des sources offre la possibilité de travailler en parallèle sur plusieurs versions du code source, de labelliser un état donné de l'ensemble d'un développement, permettant, de fait, d'identifier de manière unique le code source d'une version donnée de l'application.

Le dernier avantage du référentiel est de permettre de fusionner plusieurs versions du code source dans une nouvelle version prenant en compte le meilleur des deux versions concurrentes.

c. Serveur central unique

L'ensemble des ressources pour l'application est concentré sur un seul serveur permettant de ne pas introduire trop

© ENI Editions - All rigths reserved - Algeria Educ

de complexité autour de la plate-forme.

Le serveur d'applications aura donc les rôles suivants :

- Serveur de ressources statiques.
- Serveur d'applications.
- Serveur de messagerie.
- Serveur de centralisation des traces.
- Serveur de bases de données.
- Serveur de fichiers.
- Serveur d'analyse des codes sources.
- Serveur de tests unitaires.

Son rôle est double. Il sert de serveurs de déploiement et de validation de développement en cours. Son second rôle est de fournir l'ensemble des ressources (bases de données, serveur de messagerie, etc.) pour les développeurs afin qu'ils puissent réaliser localement leur développement et valider l'avancement de leurs fonctionnalités par des tests unitaires par exemple.

d. Banalisation du poste développeur

Le poste du développeur est le plus standard possible et ne varie que très peu d'un développeur à l'autre. Le poste n'est donc pas critique en cas de panne, il peut être substitué directement sans encombre. Une image peut être déployée automatiquement sur le poste afin de le rendre « standard » et uniforme.

3. Environnement de test en charge

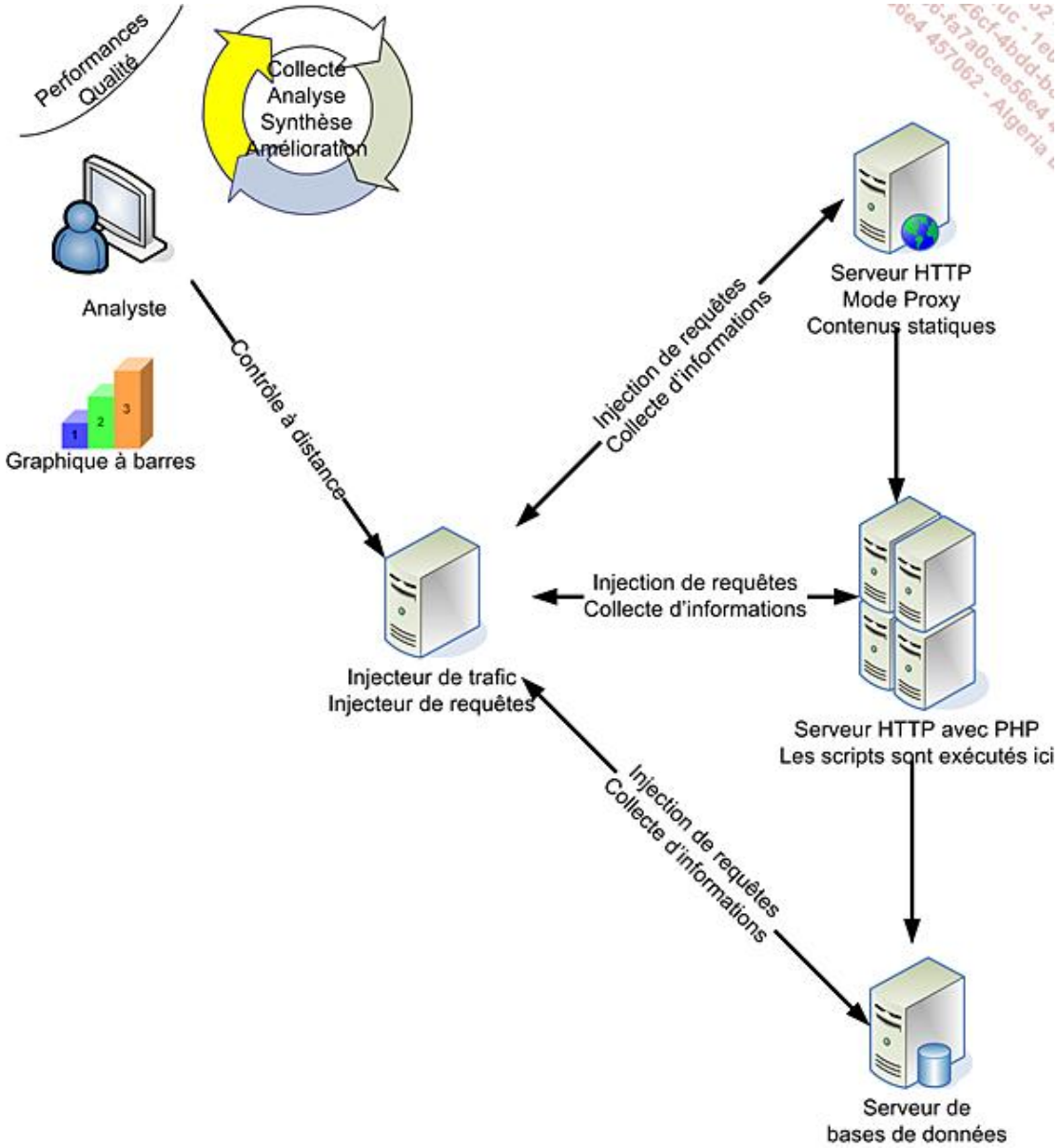
Cet environnement permet de valider la bonne tenue en charge des applications, garantissant ainsi que les applications finissant sur des plates-formes de production (recevant le trafic des clients réels) seront adaptées en matière de tenue en charge et de vieillissement.

La tenue en charge est la capacité de l'application et de l'environnement à fournir une réponse acceptable pour chaque client jusqu'à un certain seuil appelé seuil de rupture. Le seuil de rupture est un ensemble de valeurs (en matière de types de requêtes, de nombre de requêtes et de nombre de clients en parallèle) à partir desquelles l'application va se mettre à répondre plus lentement et à consommer de plus en plus de ressources sur les serveurs.

Le test de vieillissement consiste, à l'inverse, à stimuler l'application de manière modérée, soit environ 80% de charge du point de rupture. Le but étant de valider la stabilité de l'application en terme de consommation de ressources et en terme d'exactitude des résultats.

L'environnement sert donc à connaître les limites des plates-formes et à garantir la stabilité en fonctionnement nominal des applications déployées.

© ENI Editions - All rigths reserved - Algeria Educ



Cet environnement n'est mis en place que lors de l'augmentation du trafic et de la popularité des applications afin de garantir une réponse optimale et éviter les ruptures de services.

Cet environnement nécessite souvent deux ressources supplémentaires : un injecteur et le singeur.

Il est important de posséder un serveur autonome permettant de réaliser une simulation de trafic et d'injecter de nombreuses requêtes. Ce serveur agit comme un injecteur de trafic autonome. La seconde ressource nécessaire est celle qui permettra d'obtenir des réponses plus précises, de mettre le doigt sur des ressources plus critiques : le singeur. Par exemple, si vous savez que le système de cache vient masquer de nombreuses erreurs de performance, il peut être intéressant de singer ce service et de simuler des comportements extrêmes afin de mesurer l'impact des pires scénarios.

L'injecteur et le singeur ne sont pas nécessairement autonomes et peuvent être mutualisés sur des serveurs communs à l'application. De même, les opérations de désactivation ou de modification des services peuvent être utilisés en modifiant dans le code (au travers de leur configuration) les implémentations de certains services afin de déterminer quelles sont les meilleures stratégies.

4. Environnement d'intégration

Comme son nom l'indique, cet environnement sert à intégrer une application à partir de plusieurs livraisons d'applications et configurations différentes et ayant des cycles de vie autonomes.

Cet environnement a pour objectif de garantir une validation technique de l'assemblage des différentes briques.

Il a pour but de préparer la configuration pour les environnements de préproduction et de production.

© ENI Editions - All rigths reserved - Algeria Educ

La tâche la plus importante consiste à valider techniquement que la solution rend le service de base, que les paramètres sont correctement positionnés et que l'ensemble des briques ne produit plus de traces applicatives de types erreurs ou alertes.

5. Environnement de tests fonctionnels

Souvent semblable à l'environnement d'intégration après la phase de stabilisation, son but est de valider les applications d'un point de vue service et fonctionnalités au travers de cahiers de tests et de recette.

Il n'a pas pour but d'adresser des problématiques de performance (comme le fait l'environnement de test en charge) ni de gérer des problématiques d'intégration, mais de donner une vision claire du bon fonctionnement des fonctionnalités et de garantir la non-régression des fonctionnalités avec l'évolution des versions des différentes applications.

6. Environnement de préproduction

Avant-dernière étape avant le départ en production, cet environnement a pour but de valider les livraisons effectuées depuis les environnements d'intégration, avant le déploiement final en production.

Cet environnement est l'ultime passage avant le déploiement. Cet environnement ne sert pas aux tests d'intégration, ni aux tests unitaires, ni aux tests de fonctionnalités, ni aux tests en charge. Cette plate-forme sert de modèle et d'image à la version en production afin que les erreurs en production puissent être reproduites de manière autonome « en dehors du trafic » par des équipes internes.

Cet environnement n'est d'ailleurs joignable que par des équipes restreintes et ne sert que de pont de validation.

Cet environnement est la référence pour les recettes automatiques car son fonctionnement est à l'image du fonctionnement obtenu par un client et n'a pas d'impact sur le service rendu aux clients.

7. Environnement de production

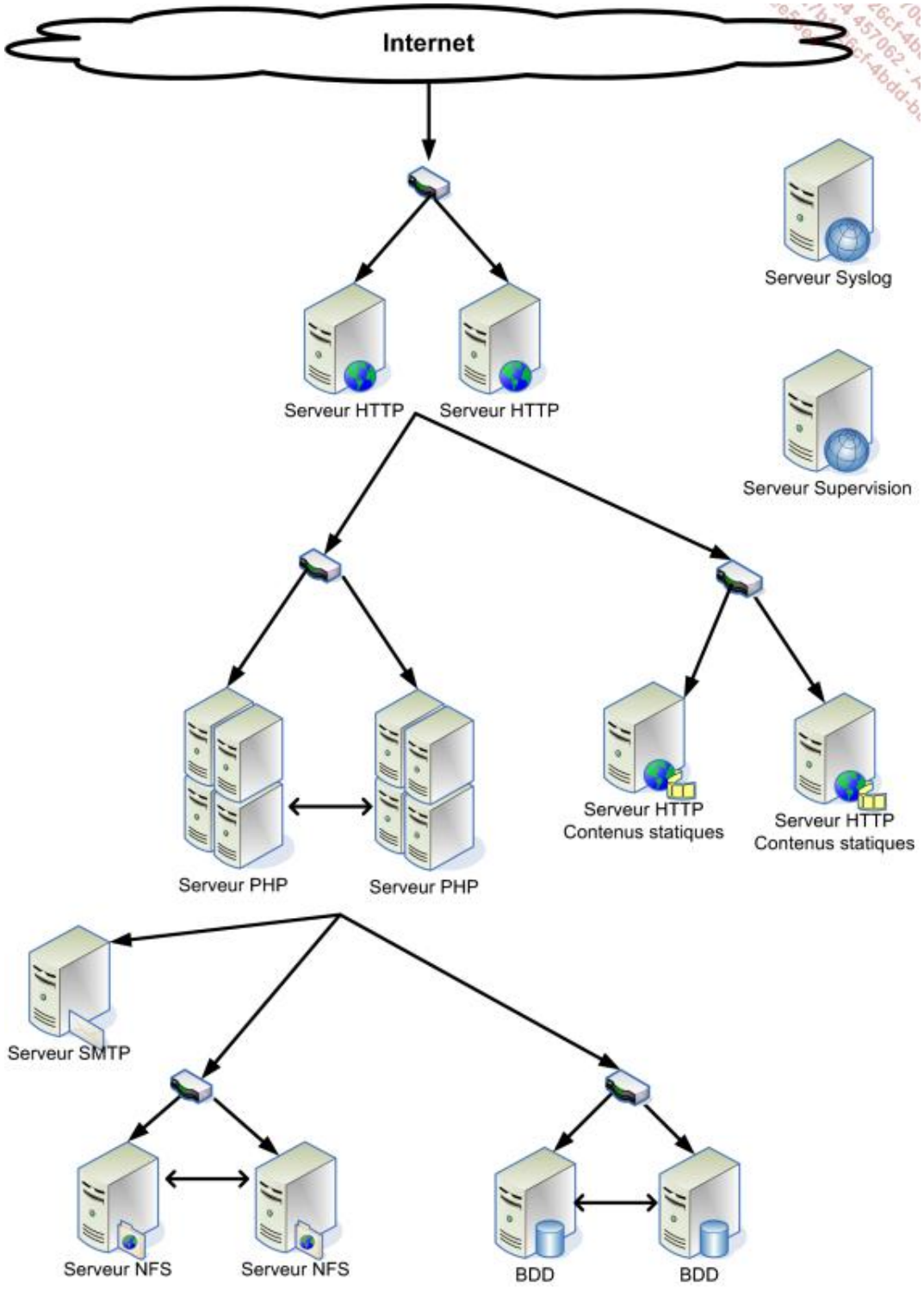
Il s'agit de la plate-forme répondant à vos clients. Elle contient des versions de vos applications considérées comme suffisamment stables pour offrir les meilleurs résultats, les meilleures fonctionnalités et les dernières nouveautés à vos clients.

Dimensionnée en conséquence, elle prend en compte les besoins réels en ressources et garantit au mieux la disponibilité des ressources.

Cet environnement possède des ressources supplémentaires telles que :

- Le superviseur : contrôlant les données techniques du service
    - Temps de réponse.
    - Consommations RAM, disque et bande passante réseau.
    - Suivi des traces.
    - Tests de fonctionnalités.
- La centralisation des logs
    - Concentration des traces sur un serveur de traces unique.
    - Format unique des messages.
    - Supervision des traces applicatives centralisées.
    - Création d'un rapport des erreurs quotidien.

© ENI Editions - All rigths reserved - Algeria Educ

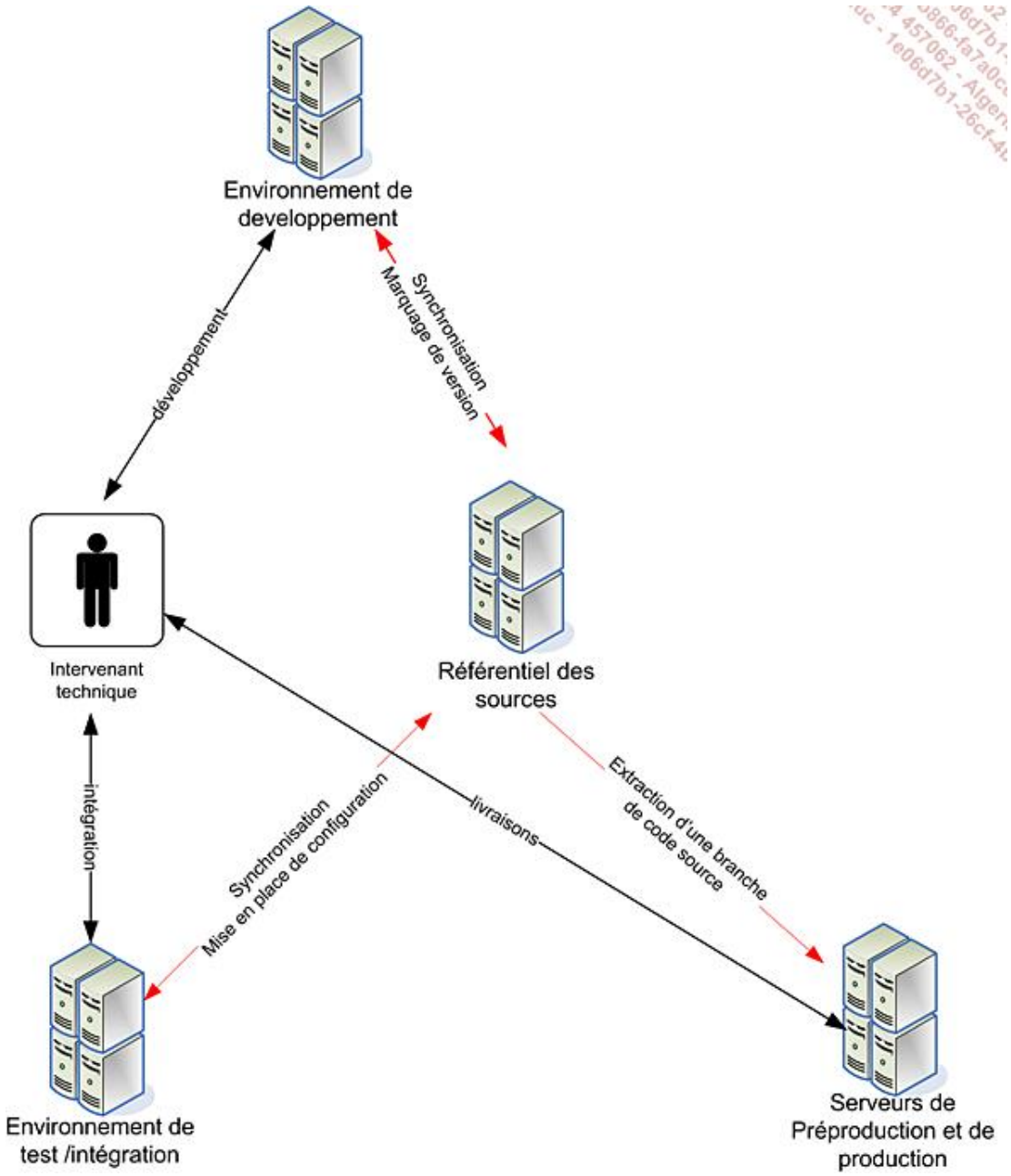


© ENI Editions - All rigths reserved - Algeria Educ

Organisation et processus interenvironnement

1. Exemple simple et efficace

Le premier exemple explore la possibilité de gérer efficacement votre code source, vos applications et vos configurations autour d'un seul serveur : le serveur de gestion des sources. Il existe plusieurs environnements qui ne sont que des « vues » de l'état des applications et des configurations.



2. Exemple complet et industriel

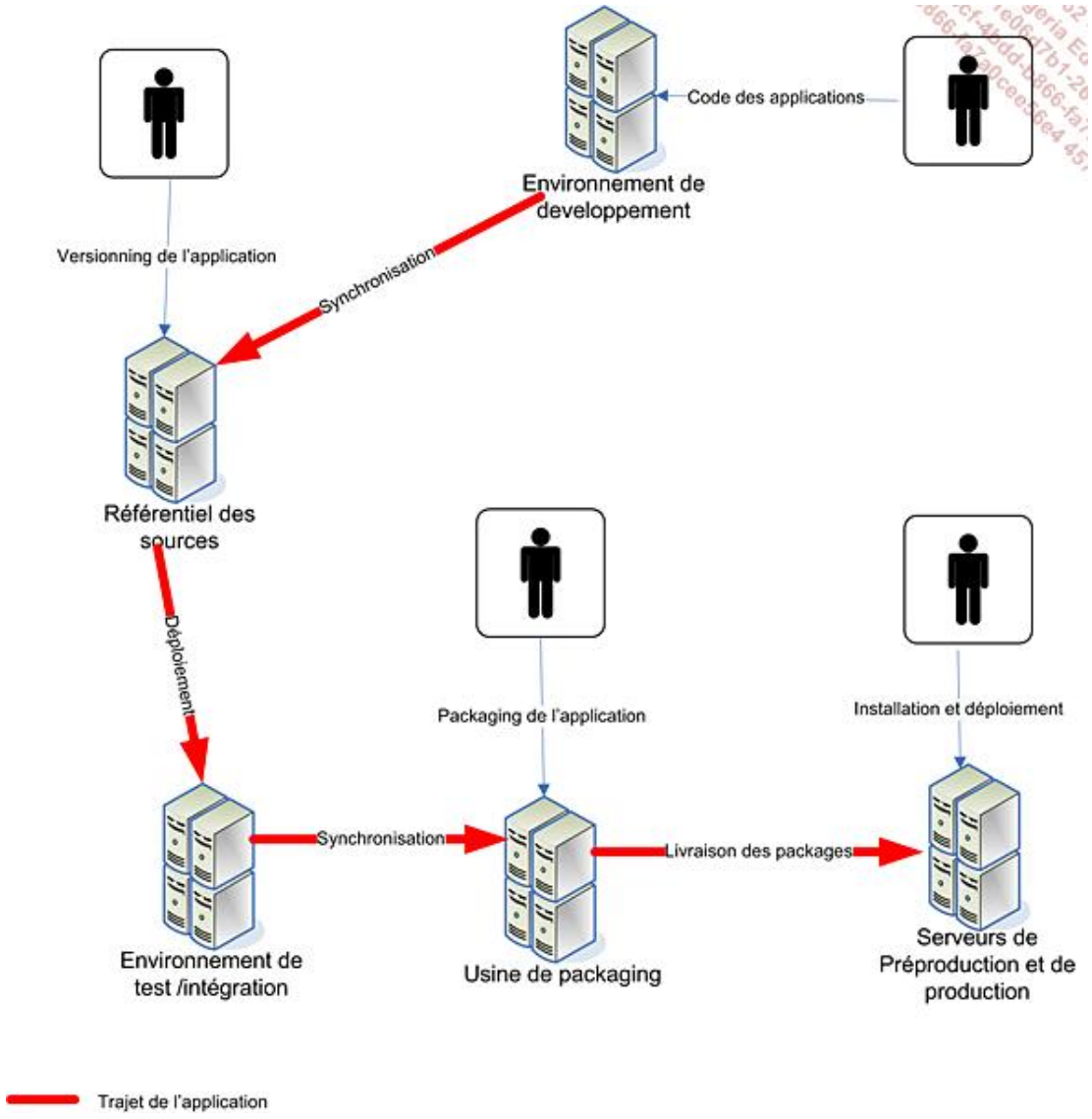
Un exemple plus complet permet à plusieurs équipes distinctes de travailler ensemble pour la mise en ligne d'une application de manière industrielle comme par exemple sur une chaîne de production.

Différentes équipes vont se succéder sur le cycle de vie et de mise en ligne de votre application :

- Équipe de développement

© ENI Editions - All rigths reserved - Algeria Educ

- Équipe d'intégration
- Équipe de tests en charge
- Équipe de tests et recette
- Équipe de déploiement
- Équipe d'exploitation



© ENI Editions - All rigths reserved - Algeria Educ

Nature d’une application en production

1. Qu’est-ce qu’une application ?

Une application fonctionnant sur une plate-forme se traduit par une équation simple.

Application = Code + Configuration

Le Code représente l’ensemble des fichiers de sources PHP contenant le comportement de l’application sous forme d’instructions.

La configuration représente l’ensemble des paramètres spécifiques au contexte technique dans lequel se trouve une application.

La configuration contient généralement :

- Les URL d’appel.
- Les paramètres de connexion aux bases de données.
- Les informations sur les caches.
- Les labels des applications, des pages, etc.
- Le langage par défaut.
- Les modules spécifiques à utiliser.
- …

La gestion d’une application fonctionnant en production réside donc dans la capacité d’une équipe ou d’un responsable à :

- Identifier le code de l’application.
- Contrôler la configuration spécifique à l’environnement.
- Valider la conformité entre la version du code et la configuration.

2. Immuabilité de la configuration

Une configuration d’application se caractérise par une stabilité dans le temps. C’est-à-dire qu’une fois les étapes d’installation, de génération ou de dépôt de la configuration franchies, celle-ci n’a pas vocation à évoluer dans le temps.

L’évolution d’élément de paramétrage suggère qu’il ne s’agit pas de configuration mais plutôt de données.

Les paramètres de configuration ne sont pas des données.

Les paramètres de configuration sont stables, les données manipulées par le code de l’application peuvent évoluer, c’est pourquoi les paramètres de configuration ne sont pas des données mais des informations environnementales stables.

Dans cette optique, il est impératif de ne jamais permettre à une application de toucher à sa configuration. Les fichiers de configuration ne doivent pas être modifiés car les risques sont importants de donner la possibilité à une application de saboter elle-même les bases de sa propre stabilité.

© ENI Editions - All rigths reserved - Algeria Educ

Modèles de configuration

1. Le cas de modèle indéfini

Ce modèle de configuration souvent répandu pour les petits projets n'inclut que peu d'individus avec des objectifs stratégiques très limités.

L'application est déployée en production sur le même environnement que le développement.

Le code est de nature volatile ou le projet n'a pas vocation à avoir un cycle de vie très important.

Dans ce cas, les valeurs des paramètres sont directement insérées dans le code de l'application.

Sans gestion de paramètre, pas d'évolution ni de déploiement simples sur plusieurs plates-formes.

D'un point de vue de la sécurité, le code étant exécuté lors de l'accès, il est donc impossible de récupérer les paramètres et les valeurs de configuration.

2. Fichier PHP et tableau de paramètres

L'idée de ce mode de configuration consiste à définir un fichier de configuration en PHP et à y déclarer un tableau PHP contenant l'ensemble des valeurs des paramètres.

Il est impératif de séparer la manière d'utiliser les paramètres du code de l'application. Donc, dans notre fichier de configuration nommé config.php, il n'y aura rien d'autre que la déclaration du contenu du tableau nommé ici $cfg.

Le fichier de configuration ne contient que des variables de configuration, pas de logique du code, même minimale.

Le fichier de configuration a des objectifs de portabilité et de compatibilité ascendante rendant l'ajout de code plus sensible.

Le fichier de configuration en PHP rend le paramétrage du serveur plus simple. En effet, il n'est plus nécessaire de sécuriser l'accès au fichier car il est impossible de récupérer le contenu. En effet, chaque accès produit l'interprétation et l'exécution du fichier de configuration rendant inaccessible son contenu. Du point de vue de la sécurité, il s'agit d'une des meilleures stratégies de paramétrage.

Exemple de fichier de configuration

```
<?php

$cfg["default"]["TITRE"]="HOME PAGE";
$cfg["default"]["CONTENU"]="Rien ne nous dit
Qu'elles vont mourir
Les voix des cigales.";

$cfg["visiteur"]["TITRE"]="PAGE DE VISITEURS";
$cfg["visiteur"]["CONTENU"]="Quelle joie
De traverser à gué la rivière en été
Sandales en main.";

?>
```

Exemple de fichier de code

```
<?php

require_once("config.php");
```

© ENI Editions - All rigths reserved - Algeria Educ

```
$page="default";
if (isset($_GET["page"]) and isset($cfg[$_GET["page"]]) )
$page=$_GET["page"];

echo "<H1>".$cfg["$page"]["TITRE"]."</H1>";
echo "<hr/><h2>contenu</h2>";
echo "<pre>".$cfg["$page"]["CONTENU"]."</pre>";
?>
```

Dans notre exemple, le fichier code contient la logique d'utilisation de la configuration. Le fichier config.php ne contient que la déclaration.

Résultat d'appel par défaut ou sans enregistrement de paramètre

```
http://localhost/php/config/index.php
http://localhost/php/config/index.php?page=inconnu

HOME PAGE
_____________________________________________________________________________
contenu
Rien ne nous dit
Qu'elles vont mourir
Les voix des cigales.
```

Résultat d'une page ayant un enregistrement de configuration

```
http://localhost/php/config/index.php?page=visiteur
PAGE DE VISITEURS
_____________________________________________________________________________
contenu
Quelle joie
De traverser à gué la rivière en été
Sandales en mains.
```

3. Fichiers INI

Le fichier INI est un format de configuration répandu dans le mode Windows et reste relativement simple à utiliser. L'API PHP offre, de plus, la méthode ultime pour son utilisation optimale : **parse_ini_file**.

Ce modèle de paramétrage ouvre des voies à des problématiques de sécurité.

Voici notre exemple, un fichier config.ini et un fichier de code index.php

L'appel à l'URL http://localhost/php/config/ini/config.ini permet la récupération de l'intégralité de la configuration.

Il est donc impératif de sécuriser son accès direct par un fichier de directive pour le serveur Apache.

Les directives peuvent être placées dans un fichier local .htaccess si votre configuration n'est pas trop restrictive ou si votre hébergeur le permet.

Sécurisation de l'accès aux fichiers ayant une extension .ini

```
<FilesMatch "\.ini$">
Order Deny,Allow
Deny from All
Allow from none
</FilesMatch>
```

Exemple de fichier de configuration

```
[default]
TITRE="HOME PAGE"
CONTENU="Rien ne nous dit
Qu'elles vont mourir
```


© ENI Editions - All rigths reserved - Algeria Educ

```
Les voix des cigales.";

[visiteur]
TITRE="PAGE DE VISITEURS";
CONTENU="Quelle joie
De traverser à gué la rivière en été
Sandales en main."
```

Exemple de fichier de code

```
<?php

$cfg=parse_ini_file("config.ini", true);
#print "<pre>".print_r($cfg, true)."</pre>";
$page="default";
if (isset($_GET["page"]) and isset($cfg[$_GET["page"]]) )
$page=$_GET["page"];

echo "<H1>".$cfg["$page"]["TITRE"]."</H1>";
echo "<hr/><h2>contenu</h2>";
echo "<pre>".$cfg["$page"]["CONTENU"]."</pre>";
?>
```

Résultat d'appel par défaut ou sans enregistrement de paramètre

```
http://localhost/php/config/index.php
http://localhost/php/config/index.php?page=inconnu

HOME PAGE
_____________________________________________________________________
contenu
Rien ne nous dit
Qu'elles vont mourir
Les voix des cigales.
```

Résultat d'une page ayant un enregistrement de configuration

```
http://localhost/php/config/index.php?page=visiteur
PAGE DE VISITEURS
_____________________________________________________________________
contenu
Quelle joie
De traverser à gué la rivière en été
Sandales en mains.
```

4. Fichier XML

Le fichier XML est un format de configuration répandu dans le mode Web et reste cependant relativement simple à utiliser. Le format XML est très verbeux et cela peut dérouter le lecteur. En effet, la structure d'un fichier XML est proche de la structure d'une page Web bien que celle-ci soit correctement formatée.

Le XML se caractérise par :

- L'utilisation de balises <Balise>…. </Balise>.
- L'utilisation d'attributs <Balise nom= »attribut1 » prenom= « attribut2 »>…
- L'encodage spécifique des 5 caractères (', «, &, <, >) dans le contenu des balises.
- La capacité de la balise à encapsuler d'autres balises : <a><c></c>.

© ENI Editions - All rigths reserved - Algeria Educ

- Toutes les sections de balises sont correctement ouvertes et fermées.

L'API PHP offre, de plus, l'extension ultime pour son utilisation optimale : **SimpleXML**.

Cette API peut être utilisée sous sa forme itérative à l'aide de la fonction `simplexml_load_file($nomdeFichier)` ou par l'API objet : `new SimpleXmlElement(file_get_contents($nomDeFichier))`.

Ce modèle de paramétrage ouvre lui aussi des voies à des problématiques de sécurité.

Voici notre exemple, un fichier config.xml et un fichier de code index.php

L'appel à l'URL http://localhost/php/config/ini/config.xml permet la récupération de l'intégralité de la configuration.

Il est donc impératif de sécuriser son accès direct par un fichier de directives pour le serveur Apache.

Les directives peuvent être placées dans un fichier local .htaccess si votre configuration n'est pas trop restrictive ou si votre hébergeur le permet.

Sécurisation de l'accès aux fichiers ayant une extension .xml

```
<FilesMatch "\.xml$">
Order Deny,Allow
Deny from All
Allow from none
</FilesMatch>
```

L'utilisation de fichiers XML est moins triviale qu'il n'y paraît car la structure du fichier XML de configuration a un impact très important sur le code de gestion des paramètres dans l'application.

Exemple de fichier de configuration

```
<?xml version='1.0'?>
<Parametres>
    <default>
        <titre>HOME PAGE</titre>
        <contenu> Rien ne nous dit
Qu'elles vont mourir
Les voix des cigales.
        </contenu>
    </default>

    <visiteur>
        <titre>PAGE DE VISITEURS</titre>
        <contenu>Quelle joie
De traverser a gue la riviere en ete
Sandales en main.
        </contenu>
     </visiteur>
</Parametres>
```

Exemple de fichier de code

```
<?php

$xml = simplexml_load_file("config.xml");
$cfg = get_object_vars($xml);

$page="default";
if (
    isset($_GET["page"]) and
    isset($cfg[$_GET["page"]] )
    )
        $page=$_GET["page"];


$titre=$cfg[$page]->titre;
$contenu=$cfg[$page]->contenu;
```


© ENI Editions - All rigths reserved - Algeria Educ

```
echo "<H1>".$titre."</H1>";
echo "<hr/><h2>contenu</h2>";
echo "<pre>".$contenu."</pre>";
?>
```

Résultat d'appel par défaut ou sans enregistrement de paramètre

```
http://localhost/php/config/index_xml.php
http://localhost/php/config/index_xml.php?page=inconnu

HOME PAGE
_____________________________________________________________________________
contenu
Rien ne nous dit
Qu'elles vont mourir
Les voix des cigales.
```

Résultat d'une page ayant un enregistrement de configuration

```
http://localhost/php/config/index_xml.php?page=visiteur
PAGE DE VISITEURS
_____________________________________________________________________________
contenu
Quelle joie
De traverser à gué la rivière en été
Sandales en mains.
```

5. Base de données

La base de données comme source des paramètres de configuration est une bonne solution afin de centraliser l'ensemble des paramètres pour l'ensemble des serveurs. Ce mécanisme peut être raffiné à l'extrême afin de prendre en compte : le serveur, la version de l'application et la validation du paramètre.

Moins triviale que les configurations sous forme de tableau PHP, fichier INI ou XML, la base de données nécessite la présence d'un serveur de bases de données opérationnel et n'ayant pas déjà de forte sollicitation. En effet, si le serveur est fortement sollicité, celui-ci est plus lent dans la fourniture des paramètres et ralentit l'accès à l'ensemble des pages ayant un besoin en paramètres de configuration.

De plus, il n'existe pas de structure intuitive, il faut la créer selon ses besoins.

Le mécanisme de gestion de la configuration nécessite une procédure d'insertion en masse de paramètres depuis un fichier afin d'accélérer la maintenance des informations de configuration.

a. Définition de la base de données

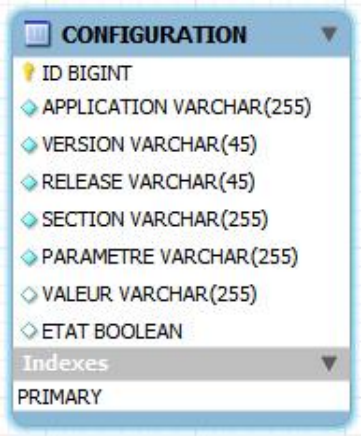
Voici donc un exemple de base de données de configuration. Vous y trouverez la description des champs, le modèle conceptuel de données et enfin le code SQL de création associé.

Nom	Type	Description
ID	BIGINT	Identifiant unique d'enregistrement
APPLICATION	VARCHAR(255)	Nom de l'application du paramètre
VERSION	VARCHAR(45)	Version de l'application
RELEASE	VARCHAR(45)	Version du livrable
SECTION	VARCHAR(255)	Nom de la section de configuration
PARAMETRE	VARCHAR(255)	Nom du paramètre

© ENI Editions - All rigths reserved - Algeria Educ

VALEUR	VARCHAR(255)	Valeur du paramètre
ETAT	BOOL	Est-il activé ou non activé ?

Modèle conceptuel de données



Code SQL de création

```
-- ----------------------------------------------------
-- Table `conf`.`CONFIGURATION`
-- ----------------------------------------------------
CREATE TABLE IF NOT EXISTS `configuration` (
  `ID` bigint(20) NOT NULL AUTO_INCREMENT,
  `APPLICATION` varchar(255) NOT NULL,
  `VERSION` varchar(45) NOT NULL,
  `RELEASE` varchar(45) NOT NULL,
  `SECTION` varchar(255) NOT NULL,
  `PARAMETRE` varchar(255) NOT NULL,
  `VALEUR` varchar(255) DEFAULT NULL,
  `ETAT` tinyint(1) DEFAULT NULL,
  PRIMARY KEY (`ID`),
  UNIQUE KEY `APPLICATION`
(`APPLICATION`,`VERSION`,`RELEASE`,`SECTION`,`PARAMETRE`)
);
```

b. Définition du composant PHP d'ajout en masse

Le but est de pouvoir insérer massivement des configurations dans notre base.

Script PHP d'insertion en masse

```
<?php

$CONFIG_FILE="./config.csv";
$DSN='mysql:host=localhost;dbname=config';
$USER='conf';
$PASSWORD='conf';

try {
    $dbh = new PDO($DSN, $USER, $PASSWORD);
} catch (PDOException $e) {
    print "Erreur !: " . $e->getMessage() . "<br/>";
    die();
}

$i=0;
$handle = @fopen($CONFIG_FILE, "r");
```


© ENI Editions - All rigths reserved - Algeria Educ

```
if ($handle) {
    while (!feof($handle)) {
        $buffer = fgets($handle, 4096);
            $elt=explode (";", $buffer);
            $REQ="INSERT INTO CONFIGURATION VALUES (NULL,
\"".$elt[0]."\", \"".$elt[1]."\",
\"".$elt[2]."\",\"".$elt[3]."\",\"".$elt[4]."\",\"".$elt[5]."\",
".$elt[6].");";
            echo "* $REQ";
            $stmt=$dbh->query($REQ);
            if (!$stmt) {
                echo "\nPDO::errorInfo():\n";
                print_r($dbh->errorInfo());
        } else {
                $i++;
        }
    }
    fclose($handle);
}
unset($dbh);
echo "*$i insertions en base.";
?>
```

Exemple de fichier de configuration config.csv

```
cms;1.0;1;default;titre;Page par defaut;1
cms;1.0;1;default;contenu;blablablablablablablablablablablabla\nbl
ablablablablablablabla;1
cms;1.0;1;default;codeCouleur;Page par defaut;0
cms;1.0;1;visiteur;titre;Page pour visiteur;1
cms;1.0;1;visiteur;contenu;VIsiteur de la et
d'ailleurs\nblablablablablablablabla;1
cms;1.0;1;visiteur;codeCouleur;Page pour visiteur;0
cms;1.0;2;default;titre;Page par defaut;1
cms;1.0;2;default;contenu;blablablablablablablablablablablabla\nbl
ablablablablablablabla;1
cms;1.0;2;default;codeCouleur;Page par defaut;0
cms;1.0;2;visiteur;titre;Page pour visiteur;1
cms;1.0;2;visiteur;contenu;VIsiteur de la et
d'ailleurs\nblablablablablablablabla;1
cms;1.0;2;visiteur;codeCouleur;Page pour visiteur;0
base;1.0;1;default;titre;Page par defaut;1
base;1.0;1;default;contenu;blablablablablablablablablablablabla\nbl
ablablablablablablabla;1
base;1.0;1;default;codeCouleur;Page par defaut;0
base;1.0;1;visiteur;titre;Page pour visiteur;1
base;1.0;1;visiteur;contenu;VIsiteur de la et
d'ailleurs\nblablablablablablablabla;1
base;1.0;1;visiteur;codeCouleur;Page pour visiteur;0
```

c. Définition du composant PHP d'interrogation

Le composant peut être représenté sous forme d'objet afin de simplifier au maximum l'initialisation.

```
<?php

class PDOConfig extends PDO {
    private $DSN='mysql:host=localhost;dbname=config';
    private $USER='conf';
    private $PASSWORD='conf';
    private $APPLI;
    private $VERSION;
    private $RELEASE;
    private $sth;
    public function __construct( $appli, $version, $release) {
        parent::__construct($this->DSN, $this->USER, $this-
```

© ENI Editions - All rigths reserved - Algeria Educ

```
>PASSWORD);
        $this->APPLI=$appli;
        $this->VERSION=$version;
        $this->RELEASE=$release;

        $REQ="SELECT VALEUR FROM configuration WHERE
APPLICATION=\"";
        $REQ.=$this->APPLI;
        $REQ.="\" AND VERSION=\"";
        $REQ.=$this->VERSION;
        $REQ.="\" AND `RELEASE`=\"";
        $REQ.=$this->RELEASE;
        $REQ.="\" AND SECTION=? AND PARAMETRE=?";
        $this->sth = $this->prepare($REQ);
    }

    public function get($section, $parametre) {
        $this->sth->execute(array($section, $parametre));
        return $this->sth->fetchColumn();


    }
}


$cfg=new PDOConfig("cms", "1.0", "1");
echo $cfg->get("default", "titre");
?>
```

Exemple de fichier de code

```
<?php

include_once("PDOConfig.php");
$cfg=new PDOConfig("cms", "1.0", "1");
$page="default";
if ( isset($_GET["page"]) )
       $page=$_GET["page"];

$titre=$cfg->get($page, "titre");
$contenu=$cfg->get($page, "contenu");

echo "<H1>".$titre."</H1>";
echo "<hr/><h2>contenu</h2>";
echo "<pre>".$contenu."</pre>";
?>
```

Résultat d'appel par défaut ou sans enregistrement de paramètre

```
http://localhost/php/config/index_pdo.php

Page par defaut

contenu
blablablablablablablablablablablabla
blablablablablablablabla
```

Résultat d'une page ayant un enregistrement de configuration

```
http://localhost/php/config/index_pdo.php?page=visiteur
Page pour visiteur

contenu
Visiteur de la et d'ailleurs
blablablablablablablabla
```


© ENI Editions - All rigths reserved - Algeria Educ

© ENI Editions - All rigths reserved - Algeria Educ

Packaging et livraisons

Afin de simplifier et normaliser la livraison des applications, le packaging et le processus de packaging sont des phases incontournables qui garantissent :

- La cohérence des livraisons entre les différents environnements.
- L'équivalence de code entre les différents environnements.
- La détectabilité des altérations de code.
- La traçabilité des modifications.

Quelles que soient la politique et la technique mises en place pour assurer le packaging et la production de livrables en environnement de production, ceux-ci doivent répondre à des critères permettant son acceptation et son adoption pour toutes vos mises en production.

Chaque livrable est autonome, ce qui rend le retour arrière possible.

Chaque production de livrables est aussi rapide ou plus rapide qu'une stratégie de déploiement « manuel » de votre application.

Chaque livrable fait référence à une version de l'application et une version de livrable permet, à tout moment, de retrouver le code source initial de chaque fichier et de comparer la version de votre référentiel de sources avec celle de la livraison.

1. Configuration et livraison

Il est important d'imaginer la configuration comme ayant deux parties distinctes :

- La configuration générale.
- La configuration spécifique.

a. Administration des configurations

Le concept d'enfer du paramétrage (config hell) est apparu il y a quelques années. Il fait référence à la gestion des configurations qui devient, si elle n'est pas maîtrisée, un véritable cauchemar pour les équipes d'intégration et de déploiement.

Il est donc impératif de considérer les fichiers de configuration au même titre que le code source de vos applications, c'est-à-dire comme des entités nécessitant une gestion dans le but d'être exploitées par d'autres équipes.

Le stockage et la labellisation des différents fichiers de configuration sont donc aussi importants que ceux du code source.

Dans la même optique, il est important de mettre en place un mécanisme de traces à chaque accès douteux vers la configuration.

```
< ?php
...
if ( !isset($cfg[$section][$parametre]) or
$cfg[$section][$parametre]==’’ or $cfg[$section][$parametre]
==null) {
    trigger_error( “[CONF]Manque Valeur pour parametre : $section
/ parametre : $parametre”, E_USER_WARNING) ;
}
...
?>
```

Ce mécanisme, bien que peu anticipatif, permet cependant de valider, en cours de fonctionnement, que l'intégralité des paramètres sont renseignés et maintient donc un certain niveau de qualité qui est clairement contrôlable,

© ENI Editions - All rigths reserved - Algeria Educ

mesurable dans les traces des applications.

b. Configuration globale

La configuration générale est étanche et globale. Celle-ci peut être directement intégrée avec chaque livraison d'application. Elle contient tout ce qui ne varie pas d'un serveur à l'autre, d'un environnement à l'autre. Cette configuration est souvent à la charge des équipes de développement qui maintiennent et font vivre cette partie de la configuration au sein même du code. Cette partie de la configuration est souvent stockée comme fichier INI ou tableau PHP. L'utilisation doit être simplifiée au maximum afin de permettre la consommation des paramètres de manière rapide et optimale.

Une configuration est spécifique par application et par version d'application. Cela signifie qu'il est important de comprendre que les configurations ne sont pas génériques et portables à souhait. Il est donc impératif de gérer les configurations de manière fine en documentant et en définissant clairement les évolutions de paramétrages entre les versions des applications.

c. Configuration spécifique

La configuration spécifique à la machine et à l'application est généralement décorrélée de la version de l'application et du livrable. Son objectif est de fournir les paramètres nécessaires à l'utilisation des ressources spécifiques à l'environnement local et de prendre en compte les ressources externes dédiées à cet environnement.

L'une des erreurs les plus courantes est d'avoir une configuration spécifique globale pour l'ensemble des applications hébergées sur le même serveur. Cela contraint l'ensemble des applications à utiliser les mêmes ressources. De ce fait, il n'est plus envisageable de partager les ressources pour des environnements différents, et ceci dans un contexte de mutualisation des ressources lié à l'apparition de matériels réseau et de serveurs de haute performance.

2. Stratégie de déploiement

Afin de garantir et sécuriser les livraisons de code, il est important de définir une procédure de livraison, la nature des livrables et la procédure de mise en place.

a. Procédure de livraison

Cette procédure doit être définie entre les équipes d'intégration et les équipes d'exploitation afin de simplifier et garantir le bon déploiement de manière optimale.

La procédure de livraison définit :

- La nature des livraisons.
- La destination des livraisons.
- L'ensemble des destinataires
- Le bordereau de livraison.
- La procédure de déploiement.

b. Nature des livraisons

Afin de mieux illustrer ce que peut être une stratégie efficace de packaging, voici cinq exemples de modèles de gestion de livrables.

- Archive Zip.
- Archive Phar.
- Paquet Rpm.

© ENI Editions - All rigths reserved - Algeria Educ

- Paquet Debian.
- Déploiement Branche subversion.

c. Procédure de déploiement

Toute procédure de livraison doit contenir la procédure de retour arrière de livraison.

L'utilisation de liens symboliques et d'un répertoire avec le nom de l'application, sa version et sa release sont d'excellents moyens de garantir un retour arrière rapide sur une livraison.

Le répertoire des livraisons peut être structuré de cette manière :

```
/livraisons/cms_1.0.1
/livraisons/cms_1.0.2
/livraisons/cms_1.1.1
/livraisons/cms_2.0.1
/livraisons/cms_2.2.1
/livraisons/monsite -> /livraisons/cms_2.2.1
```

Un retour arrière peut consister à :

- Supprimer le lien symbolique monsite
- Créer un nouveau lien monsite vers l'ancienne version cms_2.0.1

Le résultat suite au retour arrière aura donc cette forme :

```
/livraisons/cms_1.0.1
/livraisons/cms_1.0.2
/livraisons/cms_1.1.1
/livraisons/cms_2.0.1
/livraisons/cms_2.2.1
/livraisons/monsite -> /livraisons/cms_2.0.1
```

© ENI Editions - All rigths reserved - Algeria Educ

Introduction au management de la qualité

Pour améliorer grandement la qualité de vos projets, vous pouvez engager des ressources importantes dans la mise en œuvre supervisée de méthodes de qualité complexes telles que CMMI, PAQ, ITIL. En effet, leur mise en place nécessite une approche complète et totale. Ces approches ont pour but d'améliorer la gestion de la qualité et sont porteuses d'idées simples et novatrices dans le monde du Web.

La qualité dans les projets PHP a pour but d'améliorer vos productions dans le temps, et ceci, en acceptant la triste réalité que tout projet évolue et ses objectifs changent au fur et à mesure de son évolution.

La qualité doit être abordée avec un certain nombre de principes fondamentaux qui peuvent s'appliquer à la gestion de projet et qui nous servent à produire avant tout un applicatif PHP solide, robuste et capable de préparer et anticiper les évolutions et les besoins futurs.

Voici donc quelques-uns des concepts à appliquer ou plutôt dont vous inspirer afin de produire et générer le bon modèle de qualité pour votre environnement ou votre projet.

Aucune méthode ne vaut l'expérience et la compréhension.

© ENI Editions - All rigths reserved - Algeria Educ

Deux indicateurs de qualité dans vos projets

Il y a deux indicateurs plutôt stables et fiables dans l'évolution d'un projet PHP pour indiquer son niveau de qualité globale.

Le premier est la simplicité du produit. Un produit simple est souvent le résultat d'un esprit ou d'idées claires appliqués au monde technologique. Parmi les idées les plus simples, nous pouvons citer le moteur de recherche Google. L'interface initiale est très simple et intuitive et les résultats sont rapides et souvent pertinents à partir de quelques mots clés.

Le moteur de recherche Google est clairement un exemple de qualité technologique et cela se voit !

Le deuxième indicateur indiquant la présence de qualité consiste à évaluer si les équipes se focalisent sur l'essentiel et écarte toute tâche n'ayant pas une incidence directe sur les résultats ou la qualité des résultats.

Donc, si vous êtes sur la voie de la qualité, vos réalisations sont simples et intuitives. Votre démarche de réalisation est focalisée sur l'essentiel et rien d'autre.

© ENI Editions - All rigths reserved - Algeria Educ

La loi de Pareto

La loi de Pareto est une loi empirique dénommée loi des 80/20 et découverte par l'économiste Vilfredo Pareto qui lui a donné son nom.

Cette loi stipule que 80% des causes produisent 20% des résultats et que, à l'inverse, il existe 20% des causes qui produisent 80% des résultats.

Son étude a d'abord été réalisée en économie où il a montré que 80% des richesses étaient dans les mains de 20% de la population.

Cette loi est empirique car le ratio 80/20 n'a rien d'exact et peut être 90/10 ou 95/5. Cependant, ce qu'il faut retenir, c'est la présence d'un déséquilibre massif entre les causes et les conséquences.

La loi de Pareto a, par la suite, été généralisée. En voici quelques exemples :

- 20% des assurés provoquent 80% des accidents de la route.
- 20% des clients produisent 80% du chiffre d'affaire.
- 80% des dépenses de santé concernent 20% de la population.
- 20% du contenu d'une maison fait 80% de la valeur contenue.

La loi de Pareto s'applique aussi à vos projets :

- 20% du code est exécuté 80% du temps.
- 20% des fonctionnalités de votre application sont utilisées 80% du temps.
- 80% des fonctions sont réalisées pendant 20% du développement.
- 80% des utilisateurs ne connaissent que 20% des fonctionnalités.
- 20% des actions de qualité produisent 80% de la qualité.

Ce constat nous indique que nous pouvons nous lancer dans une analyse et une recherche des causes qui vont donner le gros des résultats.

Quelles sont les actions qui vont faire le plus de différence entre une application de qualité et une application médiocre ?

Choisir ce qui est important ou essentiel, selon les contextes, n'est pas toujours évident. Cependant, ce choix constitue un axe majeur de l'amélioration de la qualité de service et permet de sélectionner les évolutions qu'il faut pour votre projet pour qu'il puisse donner le meilleur usage possible.

1. Application de la loi de Pareto

L'application la plus directe de la loi de Pareto est le questionnement direct.

Qu'est-ce qui est le plus important pour avoir les meilleurs résultats indépendamment de tous les autres facteurs liés au contexte et à l'environnement ?

Il existe évidemment un ensemble très restreint d'actions permettant l'émergence d'améliorations significatives dans la qualité de vos productions en PHP. Trouvez quelles sont ces actions !

La Loi de Pareto étant généraliste et empirique par nature, il se peut que de petites modifications de l'environnement de réalisation permettent aux équipes d'être mieux dans leur travail ! Par exemple, est-ce que la climatisation est bien réglée ? Ne fait-il pas trop chaud, pas trop froid ?

La luminosité du lieu de travail est un facteur de productivité important. De nombreuses études ont prouvé que plus un espace est éclairé et plus la productivité est importante.

La mise en place d'un outil technique de partage du code, comme un serveur CVS ou subversion par exemple, va prendre entre une demi et une journée d'installation mais permettra de récupérer des heures de travail sur la fusion de code source entre les individus d'une équipe de développement car cet outil fera ce travail à leur place.

© ENI Editions - All rigths reserved - Algeria Educ

Bref, la loi de Pareto est un outil de performance indispensable pour comprendre les principes fondamentaux de la performance. La Loi de Pareto est donc un principe fondamental :

Tout n'est pas important et équivalent et un nombre réduit de causes produit le gros des résultats !

2. Méthode ABC

La méthode ABC est une méthode permettant de cibler les causes produisant la majorité des interventions afin d'améliorer la qualité dans certains domaines. Cette méthode s'appuie sur la courbe de Pareto mettant en évidence la part totale des incidents.

Il s'agit de mettre en place une mesure de l'impact des anomalies dans un tableau, puis de mettre les résultats sous forme de graphique permettant d'identifier le cœur des anomalies produisant 80% des problèmes de qualité.

a. Exemple de la courbe de Pareto

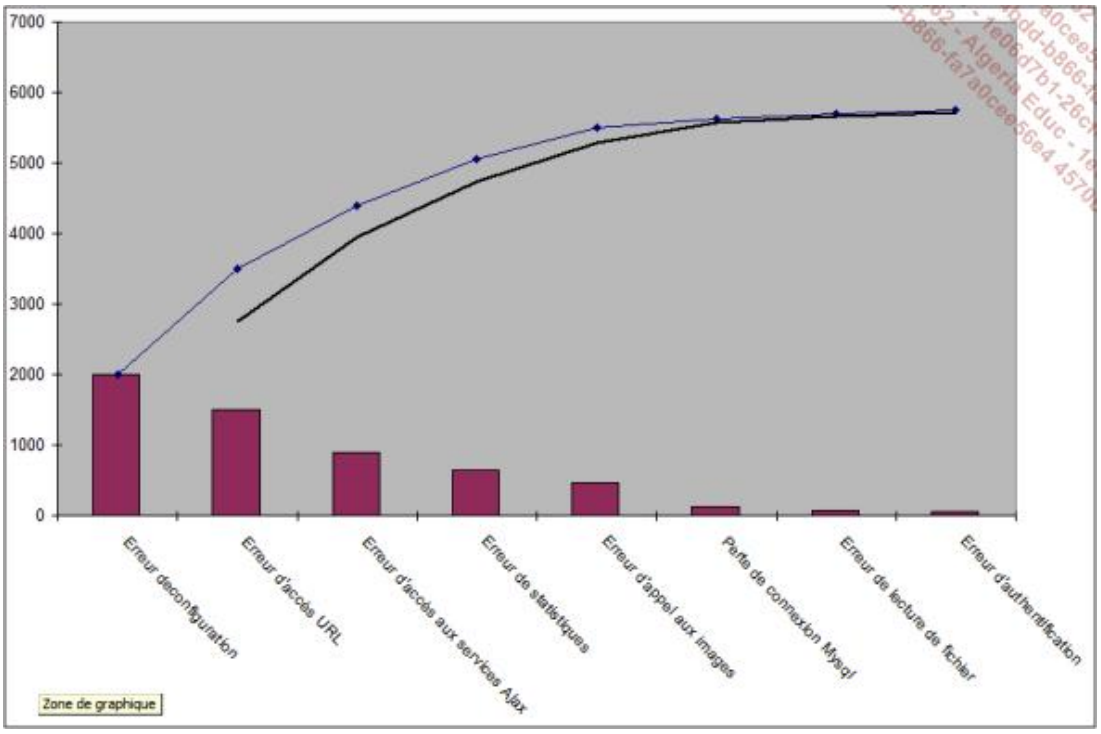
Dans le cadre d'un projet PHP mis en production depuis six mois, la récupération des erreurs générées par l'application donne les résultats suivants.

La colonne Somme contient le cumul du nombre d'erreurs. Nous allons donc chercher la ligne à partir de laquelle le cumul (colonne Somme) représente plus de 80% des erreurs. Le tableau doit, par défaut, être trié par ordre décroissant du nombre d'erreurs.

	Type d'erreur	Nombre d'erreurs	Somme	Pourcentage
1	Erreur de configuration	2000	2000	34,82%
2	Erreur d'accès URL	1500	3500	60,93%
3	Erreur d'accès aux services Ajax	900	4400	76,60%
4	Erreur de statistiques	650	5050	87,92%
5	Erreur d'appel aux images	456	5506	95,86%
6	Perte de connexion Mysql	123	5629	98,00%
7	Erreur de lecture de fichier	65	5694	99,13%
8	Erreur d'authentification	50	5744	100,00%
	Total	5744		

De ce tableau, une courbe peut être extraite, permettant de mettre en relief l'efficacité des actions.

© ENI Editions - All rigths reserved - Algeria Educ



Ce graphique représente le nombre d'erreurs dans un ordre décroissant, le pourcentage du cumul des erreurs, c'est-à-dire la part du cumul des erreurs courantes et des erreurs précédentes vis-à-vis du nombre total d'erreurs et la moyenne mobile des erreurs.

La version graphique montre que plus on est proche de la moyenne et moins l'évolution corrective offrira le maximum de potentialité et fera la différence.

Pour être efficace, il faut offrir les correctifs les plus éloignés de la moyenne mobile afin d'être sûr de l'impact des correctifs apportés à votre application.

Que doit-on faire avec ce tableau, ou ce graphique et quel est le rapport avec la loi de Pareto ?

L'idée de ce tableau est de cibler l'ensemble des erreurs dont le cumul représente plus de 80% du nombre total d'erreurs. Nous savons qu'il faudra 80% d'efforts pour corriger 20% des erreurs. Grâce à cette analyse, nous réduisons nos efforts afin d'être plus efficace en ciblant l'essentiel des problèmes et des erreurs.

Le premier point de réflexion se situe à l'issue de la correction des trois premières erreurs. En effet, la correction de ces trois erreurs, à elle seule, réduira de 76,60% exactement le volume global d'erreurs.

Le second point de réflexion se situe à l'issue de la correction des quatre premières erreurs qui produira l'élimination de 87,92% des erreurs.

Ce type d'outil est donc un « must » afin de focaliser les efforts sur les points importants en évitant le perfectionnisme. La perfection doit être vue sous l'angle de l'excellence, c'est-à-dire la recherche de l'amélioration en continu. Le fait d'éliminer le gros des problèmes permet d'avancer rapidement vers son objectif et de laisser le soin à l'avenir de décider le moment opportun du bien fondé de nouvelles corrections permettant d'améliorer encore la situation.

Il peut être appliqué à des multitudes de points importants dans votre projet PHP. Voici trois exemples parmi tant d'autres.

Le premier exemple a pour objectif d'améliorer la qualité de fonctionnement d'une application. Autant améliorer la qualité des modules ou des briques de code les plus défaillants. Les modules PHP d'où proviennent les bugs doivent donc être ciblés prioritairement avec comme indicateur mesuré le nombre de bugs par module. La solution la plus directe est le « zero bug day » ou « bug day ». Ce principe consiste à consacrer une journée à l'émergence de corrections massives sur une problématique donnée. Cette approche orientée donne de l'efficacité par la mobilisation de ressources supplémentaires pour un temps très court. Cet effort ne peut pas être maintenu dans le temps, c'est pourquoi il est court (de l'ordre de la journée) et qu'il cherche à mobiliser les ressources massivement dans une seule direction afin de permettre l'émergence de quelque chose de différent et de meilleure qualité.

Le second exemple a pour objectif d'augmenter la couverture des tests unitaires. Pour cela nous chercherons à mesurer le nombre de fonctions sans test unitaire par module. Sur un module n'ayant aucun test unitaire, le fait de réaliser une couverture totale du code dans une opération lancée avec énergie dans un temps très court produira une amélioration significative de votre qualité ou, au moins, du contrôle de cette qualité.

Le troisième exemple consiste à améliorer les temps d'exécution en ciblant les parties de traitement les plus consommatrices de temps. L'indicateur privilégié est donc le temps d'exécution du programme évalué en mesurant le temps passé par traitement. Chaque temps d'exécution des modules doit être mis en relief dans l'ensemble du traitement. Dans cet exemple, un module absorbant 6% du temps global ne pourra permettre des gains de temps

© ENI Editions - All rigths reserved - Algeria Educ

supérieurs à 6% du temps total de traitement et ceci par son élimination simple. A contrario, un traitement consommant 70% du temps de traitement peut être optimisé par l'ajout d'un cache de données, une optimisation de code ou une optimisation de transfert de données. L'amélioration de 50% de temps du module produira une amélioration directe de 35% du traitement global.

3. Émergence de standard

La loi de Pareto, permettant de se focaliser sur ce qui est important, évite de se laisser porter par l'urgence. Elle reste l'un des moyens les plus efficaces d'élever et créer de nouveaux standards pour vos applications. Effectivement, en s'appliquant scrupuleusement sur les actions qui donnent les meilleurs résultats, le niveau global de qualité s'élève et devient, de fait, le nouveau niveau de qualité standard pour vos futures évolutions ou pour vos futurs projets.

L'application de la loi de Pareto permet de se focaliser sur l'important et non sur l'urgence. L'urgence dicte ses exigences, cependant vous pouvez (par l'application du principe de « bug day ») créer quelque chose qui fera complètement la différence.

La loi de Pareto revient à se poser une question : quelles sont les actions qui feront la différence ?

© ENI Editions - All rigths reserved - Algeria Educ

La loi de Parkinson

Voici une seconde loi qui fait office, elle aussi, de principe.

La loi de Parkinson, ne provient pas du milieu médical, mais est issue d'une étude sur l'administration coloniale britannique.

La question essentielle de l'étude était de comprendre pourquoi, quand une administration embauchait du personnel, la productivité n'augmentait pas significativement en proportion des nouvelles embauches.

Les conclusions tombèrent comme des couperets : les employés créent du travail entre eux ! La masse de travail organique ou interne augmente avec le nombre d'employés.

La solution proposée a été la suppression des tâches inutiles par des managers et le recours à l'externalisation.

La Loi de parkinson est née : elle stipule que le travail s'étale de façon à occuper le temps disponible pour son achèvement.

Nous pouvons donc effectuer une généralisation de cette loi aux autres ressources utiles à l'humanité pour définir une loi de Parkinson généralisée : « Toute demande a tendance à s'étendre pour consommer toutes les ressources mises à disposition pour la réalisation. »

Nous pouvons donc trouver de nombreux exemples de notre nouveau principe :

- Les pays consomment l'énergie électrique de l'ensemble des sources de production.
- Les pays consomment l'intégralité du pétrole extrait et traité.
- Les avions sont toujours remplis même avec des places à coups réduits.
- Les consommateurs de téléphone consomment l'intégralité de leur forfait mobile.

Dans le cadre de cet ouvrage, la loi de parkinson peut indiquer la teneur de vos performances. En effet, une tâche que vous allez définir pour une journée de travail ne va pas avoir la même importance qu'une tâche définie pour un mois. Les deux tâches peuvent avoir la même finalité, cependant il y en a une qui va produire beaucoup plus de travail en matière d'étude, de prototypage, de documentation, de réunions et de points d'avancement.

Il est donc impératif de définir des tâches ayant un temps de réalisation très court afin de ne pas produire, comme une administration coloniale britannique, un surcoût de travail et de permettre à votre projet de focaliser son énergie sur l'essentiel.

Définir des temps de réalisation courts permet de focaliser l'énergie sur l'essentiel et évite la perte de temps sur des actions non productives !

Définir des tâches pour une réalisation dans un temps inférieur à quelques jours semble être un bon début et vous permettra de définir suffisamment la tâche afin d'éviter l'ambiguïté et de vous garantir la bonne formalisation de votre besoin.

© ENI Editions - All rigths reserved - Algeria Educ

Le Kaizen ou l'amélioration continue

La qualité n'est pas un lieu, une place ou un résultat, il s'agit d'un processus : le processus d'amélioration continue.

La démarche de qualité est importante. En effet, il ne s'agit pas de fournir un énorme effort pendant un temps très court afin de prétendre « faire de la qualité ».

Ce qui compte réellement est un engagement ferme inscrit dans la durée permettant d'améliorer la qualité de vos productions au fur et à mesure de la réalisation.

Kaizen est un mot qui vient du japonais. Kai signifiant « changement » et Zen signifiant quant à lui « bon ». Le Kaizen représente le bon changement et par logique l'amélioration continue car le changement est toujours présent dans chaque projet PHP.

La démarche Kaizen permet d'inciter l'équipe ou l'individu dans le projet PHP à effectuer et mener des actions quotidiennes permettant l'augmentation de la qualité dans le projet.

Les habitudes (bonnes ou mauvaises) prises au début d'un projet vont directement influencer les résultats finaux de celui-ci. Une équipe qui prend, par exemple, l'habitude de partager ses difficultés et ses savoir-faire est plus à même de résoudre des difficultés quand elles se présentent qu'une équipe n'ayant pas la volonté de partager et d'échanger régulièrement.

L'amélioration continue est un principe de base. Cependant il lui faut une méthode pratique permettant sa mise en œuvre et la validation de son bien-fondé.

Sans la volonté de faire mieux quotidiennement, la qualité ne pourra pas émerger de et dans votre projet PHP.

La volonté d'améliorer sans cesse le produit, la manière de l'obtenir, les programmes de tests unitaires ou fonctionnels, les performances... Tout cela est un cercle sans fin.

En effet, dès que le maillon faible est corrigé dans les premières étapes de la courbe de Pareto, automatiquement et implicitement, un autre élément devient à son tour le maillon faible et nécessite donc une amélioration.

Cette démarche aboutit sur le principe de qualité totale. Le zéro défaut de l'industrie. Ce principe n'est jamais atteint, cependant une application stricte de cette démarche (ou philosophie) vous y conduira.

Le niveau de qualité que vous pourrez atteindre sera, de fait, une marche sur l'escalier vous permettant d'atteindre plus sereinement le niveau supérieur.

Il n'est possible d'aller dans l'espace que lorsque l'on a réussi à voler ! Si nous ne sommes pas capables de réaliser des fonctionnalités de base à un niveau donné, pensez-vous que nous pourrons toucher la qualité totale ou le simple service de base sans avoir réussi à produire un équivalent de qualité à un niveau plus basique.

L'amélioration continue permet d'obtenir des références et des exemples permettant de justifier un niveau de qualité équivalent dans des domaines similaires et à un autre niveau d'enjeu.

Ce principe s'appelle la poussée d'excellence. Il permet d'établir de nouveaux standards d'excellence et de définir de nouvelles règles du jeu.

© ENI Editions - All rigths reserved - Algeria Educ

La méthode SMART

Nous le verrons, la roue de Deming est une méthode simple basée sur quatre phases ayant pour objectif la production de résultats rapidement. La phase de planification a pour but de fournir des informations sur l'ensemble de l'avancement du projet. Couplée avec l'équation du chien, la roue de Deming prend son sens et nous fait comprendre que le chemin le plus court vers un objectif est toujours de chercher à suivre la voie qui mène le plus rapidement aux résultats.

Il est clair qu'avec une telle technique, vous êtes propulsé dans la démarche de la « bêta perpétuelle » très chère au Web 2.0 où les fonctionnalités arrivent par itérations ainsi que les améliorations techniques.

La méthode SMART est une démarche permettant de définir clairement un ou plusieurs objectifs en mettant en relief cinq aspects particuliers de ceux-ci garantissant qu'ils sont correctement posés.

Afin de bien cadrer le sens de votre avancée vers vos objectifs "SMART", il est important de bien définir les indicateurs mettant en relief vos situations actuelles.

Avant de définir des indicateurs, il est impératif de définir clairement vos objectifs afin de sélectionner les bonnes échelles de mesure de vos résultats.

---

1. Définition des objectifs

Afin de définir un bon objectif dans le cadre SMART, l'objectif doit répondre à un certain nombre de critères :

- Spécifique
- Mesurable
- Atteignable
- Réaliste
- Temporellement défini

Nous allons maintenant présenter plus précisément chacun des points de la méthode :

L'aspect spécifique

L'objectif doit absolument être spécifique à l'environnement, au projet, aux personnes et plus globalement au contexte dans lequel il est défini. S'il est trop général ou générique, il ne sera jamais bien défini dans votre cas.

Par exemple, « je souhaite avoir un formulaire de contact en ligne permettant de contacter l'administrateur » n'a pas le même sens lorsqu'il s'agit du site d'une multinationale avec 40 filiales ou du site d'une PME locale. Il ne s'agit pas dans le premier cas de convoyer un e-mail vers une boîte unique, il s'agit aussi de mettre en place les bonnes procédures de traitement pour permettre d'atteindre les bons interlocuteurs. Cela implique des traitements supplémentaires afin de sélectionner le bon service et donc le bon mail.

L'aspect mesurable

L'objectif doit être mesurable, c'est-à-dire qu'il existe des indicateurs permettant de définir l'avancement, le niveau de progrès, la vélocité ou la mesure de la rapidité des progressions.

Plus un objectif est facilement mesurable plus il est simple de savoir si vos développements PHP vont dans le bon sens.

L'aspect atteignable

L'objectif doit être à portée de réalisation. Il y a une limite à toute chose et cela nous permet de concentrer nos efforts sur les buts réalistes. Le réalisme est nécessaire à toute personne terre-à-terre. Il permet aussi à chaque membre de l'équipe d'imaginer clairement l'atteinte de l'objectif. Il est très difficile d'imaginer quelque chose qu'il n'est pas possible d'atteindre.

L'aspect réaliste

© ENI Editions - All rigths reserved - Algeria Educ

L'objectif doit être réaliste et être un bon reflet de ce que l'on souhaite. Il doit s'insérer dans un ensemble cohérent pour être réaliste. En effet, l'objectif de coder un pacman en PHP ne vous permettra pas de réussir à produire un logiciel de comptabilité en PHP pour une entreprise du bâtiment.

L'aspect temporellement défini

Pas d'engagement sans délai. Voila le leitmotiv des partisans de mettre un délai à chaque chose. En effet, la tentation est forte de se libérer de la contrainte temps en se donnant un budget sans limite. Cependant, il est important de comprendre le temps comme une ressource vitale de votre projet PHP. Cela signifie que : pour un module PHP nécessitant 10 jours d'un développeur payé à 300 euros/jour, sans cette contrainte forte sur son temps et la gestion de son temps, un dépassement du simple au double et votre module PHP coûtera 6000 euros au lieu de 3000 euros. Ce qui fait quand même une marge importante.

2. Définition d'indicateur

Afin de définir un bon indicateur dans le cadre SMART, l'indicateur doit répondre à un certain nombre de critères qui sont proches de la définition d'objectif tout en ayant des particularités spécifiques :

- Significatif
- Mesurable
- Acceptable et accepté
- Responsable de l'indicateur
- Temporellement défini

L'aspect significatif

L'indicateur doit être significatif, c'est-à-dire qu'il est représentatif de l'évolution de la situation de manière positive ou négative dans l'atteinte de votre objectif. Son rôle est de mettre en évidence l'impact de nos actions sur l'évolution de l'indicateur et donc de la situation.

L'aspect mesurable

L'indicateur doit être mesurable sur une échelle permettant de visualiser clairement l'évolution de l'indicateur.

Ce qui peut être mesuré peut être géré.

La mesure permet de mettre en place une démarche de « benchmarking ». Cette démarche consiste à normaliser ses indicateurs pour qu'ils soient proches, ou mieux, identiques, à un autre projet similaire. Une fois cette étape de normalisation des indicateurs effectuée, la comparaison des systèmes est possible et donc la mise en relief des lacunes est facile à révéler.

L'idée est que, lorsque vous avez atteint l'excellence ou un très bon niveau de performance pour un projet, le benchmarking est l'outil idéal pour orienter et dupliquer des résultats similaires par comparaison.

L'aspect acceptable et accepté

L'indicateur, aussi bon puisse-t-il être, ne peut pas être pertinent si son rôle ou sa mesure sont contestés par des membres de l'équipe. De même, l'indicateur peut révéler par sa mesure des évolutions n'ayant pas vraiment d'adéquation avec l'évolution réelle de la situation. Par exemple, l'évolution et la prise de contrôle de territoire pour une armée est peut-être un indicateur mesurable pour estimer l'évolution de la guerre, cependant cet indicateur peut être remis en cause par le fait qu'il ne représente pas vraiment l'évolution et peut même cacher une stratégie de l'adversaire souhaitant voir s'éparpiller les troupes. Ces indicateurs comme le nombre de combattants et le niveau d'armement des deux camps peuvent être plus acceptables pour mettre en évidence l'évolution de la guerre.

De même, dans vos projets PHP, le nombre d'interfaces graphiques réalisées est souvent pris comme exemple d'indicateur permettant de mesurer l'évolution d'un projet. Cependant, l'expérience montre que la clarification des besoins, la mise en place d'un modèle de données cohérent et l'ensemble des fonctionnalités d'utilisation des données représentent le gros du travail et sont invisibles en matière de rendu visuel. Il est donc plus important de planifier les étapes de réalisation d'un projet et de mesurer le délai de réalisation, le temps consommé et le reste à faire afin d'évaluer si le projet est dans les délais ou pas.

© ENI Editions - All rigths reserved - Algeria Educ

L’aspect responsable de l’indicateur

L’indicateur n’a de sens que s’il est géré par une personne désignée comme étant responsable. En effet, sans responsable clairement désigné, les dérives d’indicateur ne trouvent pas de porteurs pour corriger celles-ci. Personne n’est désigné pour décider et mener les actions permettant d’accompagner la situation afin de corriger les écarts.

L’aspect temporellement défini

Comme pour les objectifs, un indicateur n’a de sens que pour un temps donné. La mise en place, la prise de mesure et l’analyse des résultats doivent faire l’objet d’une définition précise des délais et phases dans lesquelles elles sont valides.

© ENI Editions - All rigths reserved - Algeria Educ

Roue de Deming

La roue de Deming est un système de qualité portant le nom de son inventeur Williams Edwards Deming. Ce statisticien a mis au point une méthode de qualité simple basée sur quatre étapes fondamentales : Planifier/Faire/Vérifier/Réagir.

La roue de Deming est une implémentation directe et concrète de la démarche Kaizen ou d'amélioration continue.

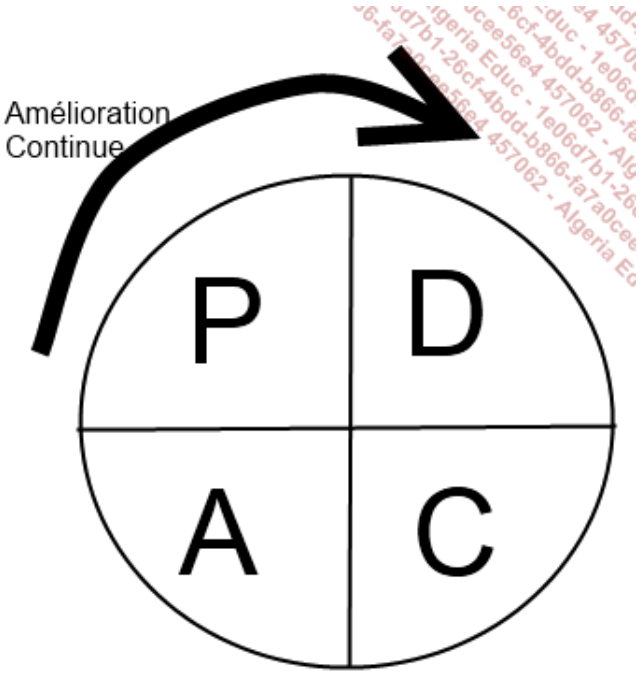
1. Étapes fondamentales

L'étape 1 (Plan) consiste à identifier un problème ou une piste d'amélioration de votre système, de bien définir sa nature, ses impacts, ses avantages, ses palliatifs et ses solutions de substitution.

L'étape 2 (Do) consiste à mener une action de mise en place d'une solution ou de réalisation d'une nouvelle fonctionnalité.

L'étape 3 (Check) consiste à mettre en place des indicateurs permettant de mesurer l'impact réel sur l'amélioration et son acceptation dans l'environnement.

L'étape 4 (Act) consiste à corriger son tir ou à réagir par rapport aux résultats obtenus par l'étape 3.



2. Planification

Il est important, voire impératif, de se fixer deux types d'objectifs afin de ne pas se noyer dans cet exercice de style :

- Des objectifs à long terme.
- Des objectifs à court terme.

Ces deux types d'objectifs vous permettent de ne jamais perdre de vue l'endroit où vous êtes, l'endroit où vous allez aujourd'hui et l'endroit où vous irez demain dans votre projet PHP.

Les objectifs à long terme permettent de définir une grande image de votre projet et de nourrir votre motivation sur le long terme en évitant de vous sentir perdu.

Les objectifs à court terme vous permettent d'avancer pas à pas et d'avoir toujours une base solide pour avancer. Il est plus facile, par exemple, de créer un site Web qui autorise le paiement d'un utilisateur quand cet utilisateur s'est déjà inscrit que quand vous n'avez qu'un site statique mis en ligne par vos soins.

Dans le cadre de la roue de Deming, le but est de mettre en place un système de gestion de la qualité par étapes et,

© ENI Editions - All rigths reserved - Algeria Educ

dans cette optique, il est important de définir des objectifs à court terme permettant de progresser et pouvant être mesurés à l'aide d'indicateurs pertinents. D'objectifs en objectifs, vous progressez vers vos objectifs plus globaux.

3. Agir et faire

Il est impératif d'aller relativement vite dans l'action afin d'avancer toujours plus vers votre objectif. Une démarche qui semble bien improbable quand on parle de qualité. Cependant, il est important de bien saisir le concept de la qualité logicielle dans son ensemble. La qualité est l'art d'offrir un service ou une application satisfaisante et répondant aux besoins dans un délai raisonnable.

Le fait d'agir vite réduira le temps de mise à disposition et réduira le coût du logiciel PHP. Le fait d'agir permet d'obtenir des résultats qui peuvent être corrigés dans le cycle suivant de la roue de Deming. Si vous n'agissez pas suffisamment rapidement, vous ne pourrez pas mesurer véritablement la qualité de votre œuvre et ne pourrez pas en déduire ni la nature ni les efforts à réaliser pour corriger votre tir.

De plus, un logiciel produit dans un temps très court n'est pas forcément de moins bonne qualité. Il m'a fallu plusieurs années de développement pour comprendre cela.

Le temps de réalisation d'un logiciel n'est pas la bonne mesure de sa qualité. Ainsi le temps n'équivaut pas à la qualité.

Les critères importants sont souvent structurels et environnementaux :

- Le niveau d'expérience des développeurs.
- La focalisation sur l'essentiel.
- Le nombre de choix techniques limité.
- L'existence de recommandations : tests unitaires, indicateurs de performance, etc.
- La motivation individuelle.
- L'organisation des équipes.
- L'entente et la communication entre individus.
- L'ambiance générale propice du projet.
- Le cadre de travail et les moyens mis à disposition.

Une bonne réalisation est une action qui ne s'étale pas dans le temps et qui est clairement définie. En effet, plus une tâche possède un délai de réalisation important plus son importance va être proportionnellement grande et plus elle va consommer de ressources. Donc, une bonne définition d'action serait de planifier des actions à 24 ou 48 heures maximum. Toute tâche définie nécessitant plus de temps doit être découpée.

Cette démarche permet d'inscrire la roue de Deming dans un cycle ultracourt permettant un pilotage au plus fin et de récupérer des résultats rapidement. Le principe de la courbe du chien est totalement respecté et offre à votre gestion de la qualité une souplesse inégalée basée sur les résultats.

4. Interprétation des résultats

a. Définition d'indicateurs

Chaque indicateur doit suivre les principes décrits par la méthode SMART afin de garantir sa consistance et son unité au sein d'un ensemble d'indicateurs.

L'indicateur est un repère normalisé permettant à chaque acteur de focaliser son attention et sa perception sur une base commune. Cela paraît surprenant au premier abord, cependant l'utilisation de ce type de démarche offre une plus grande sérénité dans la relation des acteurs et une plus grande ouverture sur les exigences et les attentes de chacun par l'analyse conjointe d'indicateurs au travers de perceptions différentes du projet PHP.

© ENI Editions - All rigths reserved - Algeria Educ

b. Mise en place d'un indicateur de qualité

La mise en place d'un indicateur doit être la moins intrusive possible.

Il est donc impératif de construire des applications et d'écrire du code supportant ce principe dès le début du projet.

Pour qu'un indicateur soit non intrusif, il faut qu'il respecte certains critères tels que :

- Bon niveau de performance.
- Désactivation sélective possible.
- Mode de collecte uniforme.
- Procédure de mise en place normalisée.
- Mise en place rapide et maîtrisée.
- Existence d'une chaîne de traitement.

En clair, il faut un processus maison ou un framework permettant de mesurer et de tirer de vos réalisations PHP les informations sur ses niveaux de performance, d'exactitude, de volumétrie de données et de complexité.

Il est impératif de positionner des indicateurs jusqu'à ce que vous ayez un sentiment authentique de compréhension et de connaissance de votre application dans ses aspects positifs et négatifs.

Votre niveau de compréhension et de maîtrise vous permettra de garantir, preuve à l'appui, que votre application peut atteindre le prochain seuil de performance.

La connaissance des performances vous permettra d'anticiper les évolutions de l'architecture ou de votre application afin de ne pas vous retrouver au pied du mur.

La mise en place d'indicateurs est le bon moyen de garantir l'avenir de vos applications.

Nous allons donc voir comment mettre en place un système de collecte personnel et efficace. Les résultats pourront être exploités sous Excel ou OpenOffice, par exemple.

Le système doit absolument être simple et efficace. La version suivante permettra de produire des graphiques à partir des informations collectées et la troisième version vous offrira la possibilité de générer automatiquement toutes les informations chaque nuit ou chaque semaine. Cette troisième version vous servira de base pour réaliser l'intégration continue de vos produits et offrir de l'agilité dans votre capacité à livrer des évolutions rapidement tout en garantissant un seuil minimum de qualité de votre livraison.

c. Collecte de l'information

Nous allons expliciter un mécanisme technique simple utilisant des fichiers texte au format CSV permettant de regrouper de manière centralisée les différents résultats de la collecte.

Comme prévu, nous mettrons en place un système de désactivation sélective de la collecte.

Le système doit être mis en place dès le début du projet afin de permettre une systématisation des mises en place.

La collecte doit permettre une analyse par production de graphiques sous Microsoft Excel ou Open Office.

Ce type de collecte est présenté dans le chapitre sur la gestion de la qualité de code.

d. Analyse et synthèse

Il est important de mettre en relief l'ensemble des indicateurs et de mettre en évidence le niveau réel d'aboutissement d'un projet. En effet, il est tout à fait envisageable de suivre un projet par l'évolution d'indicateurs multiples afin de visualiser le niveau d'aboutissement d'un projet.

Pour permettre une évolution rapide d'un produit technologique dans l'esprit du Web actuel, il est impératif de

© ENI Editions - All rigths reserved - Algeria Educ

focaliser les efforts vers ce qui donne les meilleurs résultats.

© ENI Editions - All rigths reserved - Algeria Educ

La courbe du chien

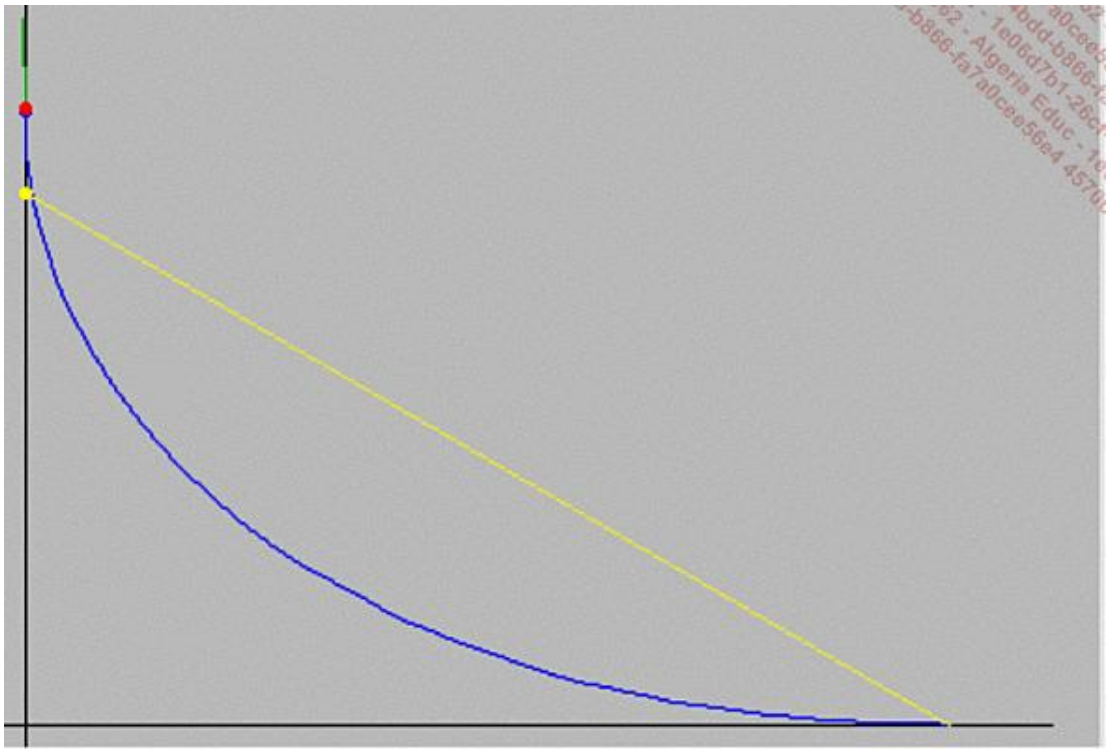
La question est : comment mettre en place un système permettant de faire la différence en fonction des évolutions ?

La piste majeure consiste à contrôler l'orientation des actions et du sens des tâches à réaliser.

La courbe du chien est idéale car ce système, bien que très simple, s'apparente au mécanisme des missiles à têtes chercheuses. En fonction des résultats des précédentes actions, on choisit l'orientation des actions immédiates à réaliser.

La roue de Deming est un environnement permettant de favoriser l'approche courbe du chien. En mathématiques en effet, à chaque progression la trajectoire est recalculée afin de correspondre à chaque fois à la meilleure stratégie du moment. La direction à prendre dans le mouvement correspond à la direction menant à la cible à cet instant.

D'étape en étape, avancer dans la progression de la qualité du système permet d'amener l'ensemble du système, pas à pas, à un très haut niveau de qualité.



Lorsque les objectifs évoluent dans le temps, l'équation du chien et son principe d'orientation dynamique vers la cible reste le moyen de s'assurer de parcourir le plus court chemin vers ces objectifs. Ceci reste vrai uniquement dans le cas où les objectifs ne sont pas fixes mais dynamiques et soumis à l'évolution dans le temps. Dans cet exemple, le point mobile correspondant au lièvre (ou au maître) se déplace à vitesse constante et uniquement sur les ordonnées.

Dans la vie d'un projet, le déplacement de la cible est moins linéaire, sa vitesse non constante, cependant cette stratégie consistant à modifier son chemin à chaque fois que l'on a franchi une étape est le meilleur moyen d'être toujours dirigé vers un chemin qui rapproche de son but.

© ENI Editions - All rigths reserved - Algeria Educ

Combinaison des principes et des démarches

La recette consiste à définir les objectifs finaux avec la méthode SMART afin d'avoir une cible clairement définie.

Dès que ces objectifs sont définis, il faut les décomposer en tâches réalisables en un minimum de temps selon le principe de la loi de Parkinson. Des tâches à la journée sont un bon début.

Il est impératif d'y ajouter l'ensemble des actions permettant de faciliter, d'améliorer et de simplifier l'organisation et la production de votre application PHP.

Dès la définition et le découpage réalisés, il est impératif d'appliquer la loi de Pareto permettant de répondre à la question : qu'est-ce qui va permettre à mon applicatif de voir le jour le plus rapidement possible et me permettre d'avoir un résultat à montrer régulièrement, faisant office de preuve « vivante » de l'avancement du projet ?

Dès la priorisation des tâches effectuées selon ce classement, il suffit de les mettre en actions dans le cycle de la roue de Deming (PDCA). Il suffit de faire tourner la roue.

À chaque fin de cycle, une analyse rigoureuse des retours doit être effectuée afin de prioriser à nouveau, d'ajouter les nouvelles tâches et d'éliminer les tâches qui sont devenues obsolètes par effet de bord des actions menées. L'idée majeure consiste donc à appliquer le principe de la courbe du chien qui adapte sa direction en fonction des événements et de l'orientation de la situation.

© ENI Editions - All rigths reserved - Algeria Educ

En résumé

La qualité est un processus et non une tâche inutile. Son cycle doit être très court pour être efficace. La qualité est un processus continu appelé amélioration continue. Sans volonté d'améliorer quotidiennement votre qualité, votre projet PHP finira toujours par souffrir de problèmes que vous devrez pallier dans l'urgence à la fin du projet. Il existe toujours quelques actions qui amélioreront massivement la qualité. Trouvez les actions qui améliorent la qualité à peu de frais. Voyez petit et à court terme pour ne pas vous perdre dans la grandeur des idées, les espoirs lointains et les incertitudes de résultats ultérieurs.

© ENI Editions - All rigths reserved - Algeria Educ